

UNIVERSIDAD POLITÉCNICA DE MADRID

FACULTAD DE INFORMÁTICA



Ph. D. Dissertation

OntoTag
A Linguistic and Ontological
Annotation Model
Suitable for the Semantic Web

AUTHOR: D. Antonio Pareja-Lora
DIRECTOR: D^a Guadalupe Aguado de Cea
CO-DIRECTOR: D^a Asunción Gómez-Pérez

‘Somebody has been working a lot’

Charles J. Fillmore [pointing to OntoTag’s ontologies]

28th May 2004 (LREC 2004)

Lisbon, Portugal

**Words are,
in my not-so-humble opinion,
our most inexhaustible source of magic.
Capable of both inflicting injury, and remedying it.**

Prof. Albus Dumbledore,
in J.K. Rowling’s
Harry Potter and the Deathly Hallows

1. INTRODUCTION.....	9
1.1. LINGUISTIC TOOLS AND ANNOTATIONS: THEIR LIGHTS AND SHADOWS	9
1.1.1. <i>The Lights: Linguistic Tools And Annotations Are Very Useful...</i>	9
1.1.2. <i>The Shadows Of Linguistic Tools And Annotations</i>	10
1.1.2.1 Linguistic Tools Are Usually Expensive.....	10
1.1.2.2 Sometimes, Linguistic Tools And Annotations Are Not Accurate	11
1.1.2.3 Linguistic Tools And Linguistic Annotations Hardly Interoperate	11
1.2. THE PROBLEM OF LINGUISTIC TOOLS AND/OR ANNOTATIONS INTEROPERATION IN DETAIL	11
1.3. SOLVING THE PROBLEM: AN OUTLINE OF OUR PROPOSAL	15
1.3.1. <i>The Role Of Ontologies</i>	15
1.3.2. <i>The Role Of Standardisation</i>	16
1.3.3. <i>A Desideratum: Minimise Cascaded Errors</i>	16
1.4. OUR PROPOSAL: MAIN PILLARS, CONTRIBUTIONS AND RESULTS	17
1.5. STRUCTURE OF THIS DISSERTATION	18
2. WORK OBJECTIVES	19
2.1. OPEN RESEARCH PROBLEMS	19
2.2. GOALS OF THE PRESENT WORK.....	20
2.3. EXPECTED CONTRIBUTIONS AND RESULTS	24
2.4. WORK ASSUMPTIONS, HYPOTHESES AND RESTRICTIONS.....	25
2.4.1. <i>Assumptions</i>	26
2.4.2. <i>Hypotheses</i>	27
2.4.3. <i>Restrictions</i>	28
3. STATE OF THE ART	31
3.1. ANNOTATION: A HISTORICAL APPROACH AND BASIC TERMINOLOGY	32
3.2. ANNOTATION: CURRENT APPROACHES	36
3.2.1. <i>The Linguistic Approach to Annotation</i>	37
3.2.1.1 Morphosyntactic Annotations.....	37
3.2.1.2 Syntactic Annotations.....	40
3.2.1.3 Semantic Annotations.....	41
3.2.1.3.1 Semantic Annotation Layers	42
3.2.1.3.2 Semantic Annotation-Related Projects, Guidelines and Standardisation Initiatives.....	44
3.2.1.3.2.1 Regarding the Sense Tagging Layer.....	45
3.2.1.3.2.2 Regarding the Semantic Domain Annotation Layer.....	49
3.2.1.3.2.3 Regarding the Semantic Field Annotation Layer	51
3.2.1.3.2.4 Regarding the Semantic Role Labelling Layer.....	52
3.2.1.3.3 Semantic Annotations – Concluding Remarks	53
3.2.1.4 Linguistic Annotation: Level-Independent Approaches.....	54
3.2.2. <i>The Computational Approach to Annotation</i>	58
3.2.2.1 Computational Representation of Annotations: Annotation Languages	58
3.2.2.2 The Semantic Web and Semantic (Web) Annotations	62
3.2.2.2.1 Definition of Ontological Tagsets and Metadata: RDFS and OWL	67
3.3. CONCLUDING REMARKS: GUIDELINES FOR A HYBRID APPROACH TO ANNOTATION	69
4. ONTOTAG: THE HYBRID ANNOTATION MODEL	71

4.1.	ONTOTAG'S LINGUISTIC ONTOLOGIES	74
4.1.1.	<i>Building Ontologies with METHONTOLOGY</i>	75
4.1.2.	<i>The OntoTag Integration Ontology (OIO)</i>	78
4.1.2.1	OIO Glossary of Terms.....	79
4.1.2.2	OIO Concept Taxonomies	88
4.1.2.3	OIO <i>Ad Hoc</i> Relationships.....	90
4.1.2.4	OIO Concept Dictionary	92
4.1.2.5	OIO Detailed Tables	94
4.1.2.5.1	OIO <i>Ad Hoc</i> Binary Relation Table	94
4.1.2.5.2	OIO Instance Attribute Table	95
4.1.2.5.3	OIO Class Attribute Table.....	97
4.1.2.5.4	OIO Constant Table.....	97
4.1.2.6	OIO Formal Axioms and Rules	98
4.1.2.6.1	OIO Rule Table	98
4.1.2.6.2	OIO Formal Axiom Table	98
4.1.2.7	OIO Instance Table.....	101
4.1.2.8	The OIO Statistics.....	102
4.1.3.	<i>The Linguistic Unit Ontology (LUO)</i>	103
4.1.3.1	The Concept Linguistic Unit, its Subclasses and its Attributes	103
4.1.3.2	The Syntactic Module	105
4.1.3.2.1	The Syntactic Module Concepts and Taxonomy.....	106
4.1.3.2.2	The Syntactic Module Attributes	114
4.1.3.2.3	The Syntactic Module <i>Ad Hoc</i> Relations	116
4.1.3.2.4	The Syntactic Module Rules and Axioms	116
4.1.3.3	The Semantic Module	120
4.1.3.3.1	The Semantic Module Concepts and Taxonomy.....	120
4.1.3.3.2	The Semantic Module Attributes	129
4.1.3.3.3	The Semantic Module <i>Ad Hoc</i> Relations	131
4.1.3.3.4	The Semantic Module Rules and Axioms	132
4.1.3.4	The LUO Statistics.....	137
4.1.4.	<i>The Linguistic Attribute Ontology (LAO)</i>	138
4.1.4.1	The LAO Concepts, Taxonomy and Instances	139
4.1.4.1.1	Top-Level Concepts and Taxonomy in the LAO	140
4.1.4.1.2	Syntactic Concepts, Taxonomy and Instances in the LAO	141
4.1.4.1.3	Semantic Concepts, Taxonomy and Instances in the LAO	144
4.1.4.2	The LAO Attributes	146
4.1.4.3	The LAO <i>Ad Hoc</i> Relations	147
4.1.4.4	The LAO Rules and Axioms.....	147
4.1.4.5	The LAO Statistics.....	150
4.1.5.	<i>The Linguistic Value Ontology (LVO)</i>	150
4.1.5.1	The LVO Concepts, Taxonomy and Instances	151
4.1.5.1.1	Top-Level Concepts and Taxonomy in the LVO	151
4.1.5.1.2	Syntactic Concepts, Taxonomy and Instances in the LVO	152
4.1.5.1.3	Semantic Concepts, Taxonomy and Instances in the LVO	160
4.1.5.2	The LVO Attributes	162

4.1.5.3	The LVO <i>Ad Hoc</i> Relations.....	163
4.1.5.4	The LVO Rules and Axioms	163
4.1.5.5	The LVO Statistics	168
4.2.	ONTOTAG'S ABSTRACT ANNOTATION ARCHITECTURE.....	169
4.2.1.	<i>Phase 1 – Distillation</i>	171
4.2.2.	<i>Phase 2 – Tagging</i>	172
4.2.3.	<i>Phase 3 – Standardisation</i>	172
4.2.4.	<i>Phase 4 – Decanting</i>	173
4.2.5.	<i>Phase 5 – Merging</i>	175
4.2.5.1	Sub-Phase 5.1 – Combination (Intra-Level Merging)	175
4.2.5.1.1	L+POS Combination.....	177
4.2.5.1.2	POS+M Combination.....	178
4.2.5.1.3	Syntactic Combination.....	179
4.2.5.1.4	Semantic Combination.....	181
4.2.5.2	Sub-Phase 5.2 – Integration (Inter-Level Merging).....	181
4.2.5.3	Putting Combination and Integration Together	182
4.3.	ONTOTAG'S ABSTRACT ANNOTATION SCHEME.....	186
4.3.1.1	Morphosyntactic Annotations.....	193
4.3.1.2	Other Annotations at the Syntactic Level.....	199
4.3.1.3	Annotations at the Semantic Level.....	203
4.3.1.3.1	Annotations at the Concept Semantic Annotation Layer	204
4.3.1.3.2	Annotations at the Instance Semantic Annotation Layer	206
5.	ONTOTAGGER: AN INSTANCE OF ONTOTAG	209
5.1.	LINGUISTIC ANNOTATION TOOLS INTEGRATED INTO ONTOTAGGER	209
5.1.1.	<i>Connexor's FDG (Functional Dependency Grammar) Parser for Spanish</i>	211
5.1.2.	<i>Bitext's DataLexica</i>	213
5.1.3.	<i>LACELL's POS tagger</i>	215
5.2.	ONTOTAGGER'S CONFIGURATION COMPONENTS	216
5.3.	THE COMBINATION MODULE (MMACM) IN DETAIL	221
5.3.1.	<i>Syntactic Structure Combination</i>	222
5.3.2.	<i>Morphosyntactic Category and Lemma Combination</i>	225
5.3.2.1	Morphosyntactic Category Combination.....	226
5.3.2.1.1	Mathematical Terms Applied in the Notation.....	226
5.3.2.1.2	Description of the Notation.....	227
5.3.2.1.3	Morphosyntactic Category Combination Rules	229
5.3.2.2	Lemma Combination	237
5.3.2.2.1	Mathematical Terms Applied in the Notation.....	237
5.3.2.2.2	Description of the Notation.....	237
5.3.2.2.3	Lemma Combination Rules.....	239
5.3.3.	<i>Morphological Combination</i>	244
5.3.3.1	Mathematical Terms Applied in the Notation	244
5.3.3.2	Description of the Notation	246
5.3.3.3	Morphological Combination Rules	249
5.4.	THE SEMANTIC ANNOTATION MANAGER MODULE (SAMM) IN DETAIL	257

5.4.1.	<i>OntoTaggerLex – The Semantic Lexicon</i>	262
5.4.2.	<i>The Instance Semantic Annotation Layer</i>	267
5.4.2.1	Named Entity-Related Linguistic Heuristics	270
5.4.2.1.1	Description of the Notation	270
5.4.2.1.2	Rules for Named Entity Recognition	274
5.4.2.1.2.1	Rules for Simple Named Entity Identification.....	274
5.4.2.1.2.2	Rules for MultiWord Named Entity Aggregation.....	275
5.4.2.1.3	Rules for Named Entity (Sub)Classification	278
5.4.3.	<i>The Concept Semantic Annotation Layer</i>	285
5.4.3.1	Domain-Independent Sense Tagging: <i>EuroWordNet</i> Integration	285
5.4.3.2	Domain-Dependent Sense Tagging: CNEO Integration	286
5.4.4.	<i>Putting Semantic Annotations Together: the Semantic Intra-Level Merger (SILM)</i>	289
5.4.5.	<i>The Semantic Learning Process</i>	290
5.5.	ONTOTAGGER’S ANNOTATION SCHEMAS.....	291
5.5.1.	<i>XML Implementation: an Example</i>	291
5.5.2.	<i>OWL Implementation: an Example</i>	301
6.	RESULTS AND EVALUATION	309
6.1.	EVALUATION OF THE COMBINATION SUB-PHASE	310
6.1.1.	<i>Statistical Analysis of the Morphosyntactic Category Combination Results</i>	312
6.1.1.1	Precision-related (generic) statistical indicators	312
6.1.1.2	Recall-related (particular) statistical indicators	314
6.1.2.	<i>Statistical Analysis of the Lemma Combination Results</i>	317
6.1.3.	<i>Statistical Analysis of the Morphological Attribute Combination Results</i> ...	319
6.2.	EVALUATION OF THE NAMED ENTITY RECOGNITION AND CLASSIFICATION SUBSYSTEM.....	322
7.	CONCLUSIONS	325
7.1.	MAIN OUTCOMES	325
7.2.	OTHER PRACTICAL OUTCOMES	327
7.2.1.	<i>Recommendations, Best Practices and Lessons Learned For Annotation Standardisation, Interoperation And Merge</i>	328
7.3.	OTHER THEORETICAL OUTCOMES	331
7.3.1.	<i>Concerning Standardisation</i>	331
7.3.2.	<i>Concerning Annotation Interoperation And Merge</i>	332
7.4.	ONTOTAG’S HYPOTHESES ASSESSMENT	333
8.	FURTHER WORK	337
9.	ACKNOWLEDGEMENTS	341
10.	REFERENCES	347
11.	APPENDIX A: ONTOTAGGER’S ANNOTATION XML SCHEMA	359

1. INTRODUCTION

Computational Linguistics is already a consolidated research area. It builds upon the results of other two major research areas, namely Linguistics and Computer Science and Engineering, and it aims at developing computational models of human language (or natural language, as it is termed in this area). Possibly, its most well-known applications are the different tools for processing human language developed so far, such as machine translation systems and speech recognizers or dictation programs.

1.1. LINGUISTIC TOOLS AND ANNOTATIONS: THEIR LIGHTS AND SHADOWS

These tools for processing human language are commonly referred to as linguistic tools. Apart from the examples mentioned above, there are also other types of linguistic tools that perhaps are not so well-known, but on which most of the other applications of Computational Linguistics are built. These other types of tools comprise POS taggers, natural language parsers and semantic taggers, amongst others.

1.1.1. THE LIGHTS: LINGUISTIC TOOLS AND ANNOTATIONS ARE VERY USEFUL...

Basically, **POS taggers** (1) segment an input text into its constituent words; (2) attach to each of these words their corresponding grammatical category (their part-of-speech or, abbreviated, their POS tag, namely noun, verb, adverb, adjective, etc.); and (3) usually supplement these POS tags with the morphological attributes that characterise the word to which they are attached (*i.e.*, its grammatical gender and number, for example).

Natural language parsers find out (1) how the words in the input text are combined and/or related to each other in order to build up more complex linguistic units, such as phrases, clauses and sentences, or (2) which syntactic role (subject, object, agent, etc.) play these words in these more complex linguistic units (or, equivalently, the dependencies that hold between these words).

Lastly, **semantic taggers** seek to assign to each word (or group of words, as the most complex semantic taggers do) (1) its particular sense or (2) its semantic role (agent, patient, source, instrument, etc.) within the sentence to which they belong.

These three types of linguistic tools, as well as the information they supply, are key to determining the meaning of an expression automatically (that is, by means of a computer, with no human intervention). For example, for an English speaker, the differences between the expressions (i) ‘the assistant professor’, (ii) ‘the assistant’s professor’ and (iii) ‘the professor’s assistant’ would be fairly clear. However, for these differences to be so clear for a computer, the computer needs the information supplied by these types of tools. Indeed, the computer needs to know, at least, (a) the meaning of the words ‘professor’ and ‘assistant’ (*i.e.*, their semantic tags); (b) how the global meaning of the corresponding expression is affected by their relative ordering (*i.e.*, their syntactic parsing); (c) that «'s» is a possessive case mark (*i.e.*, their POS tagging); and (d) how it modifies the meaning of the word to which it is adjoined (*i.e.*, their semantic role labelling). Without this information, the computer would understand these expressions as much as a person who did not know a word of English.

Therefore, linguistic tools are important assets. In fact, POS and semantic taggers (and, to a lesser extent, also natural language parsers) have become critical resources for the computer applications that process natural language. Hence, any computer application that has to analyse a text automatically and ‘intelligently’ will include at least a module for POS tagging. The more it needs to ‘understand’ the meaning of the text it processes, the more linguistic tools and/or modules it will incorporate and integrate.

1.1.2. THE SHADOWS OF LINGUISTIC TOOLS AND ANNOTATIONS

However, in general, linguistic tools (1) are expensive to develop and/or to purchase; (2) have an associated error rate, which is not always so low as desired; and (3) are difficult to integrate together into other applications (*i.e.* they hardly interoperate).

1.1.2.1 LINGUISTIC TOOLS ARE USUALLY EXPENSIVE

The expensiveness of linguistic tools results directly from the nature of the task they carry on, that is, modelling and/or processing natural language. The exceeding complexity of human language makes modelling and/or processing it exceedingly complex too. Accordingly, any tool that aims at processing natural language will be complex *per se*, and developing it will take a long time. Obviously, this increases the cost of production of linguistic tools, and this high cost of production is passed on to the customers. The customers, in turn, have to use these tools extensively in order to recoup their cost.

1.1.2.2 SOMETIMES, LINGUISTIC TOOLS AND ANNOTATIONS ARE NOT ACCURATE

Whereas the alleged error rate of linguistic tools varies between 5 and 25 percent, it actually varies between 10 and 50 percent when tested in the laboratory (see sections 5.1 and 0, and also the results of the SemEval-2 tasks¹). It depends on, mainly, (a) the kind of linguistic tool being tested and/or the technology used to develop it, and (b) the language being processed. Thus, on the one hand, POS taggers and annotations usually have an error rate ranging between 85% and 90%, whilst the minimum error rate of semantic taggers and annotations is 75%. Yet, the latter error rate increases drastically when the testing conditions (such as the language or the domain being analysed) are changed. On the other hand, the linguistic tools developed for English usually yield better error rates than those developed for more complex languages, such as Chinese, Turkish and, to a lesser extent, Spanish. Besides, the error rate of the linguistic tools developed for a language is usually inversely proportional to the number of tools developed for that language and the amount of time and human resources appointed to their development. For this reason, it is usually very difficult to find appropriate linguistic tools for processing minority and/or endangered languages.

1.1.2.3 LINGUISTIC TOOLS AND LINGUISTIC ANNOTATIONS HARDLY INTEROPERATE

As regards the integration of linguistic tools and/or annotations, the experiments carried out so far have shown that it is a pretty hard and difficult task, basically due to their multiple particularities and complex combination possibilities. Using a more recent computational term, it could be said that linguistic tools and/or annotations scarcely interoperate. That is, when trying to integrate and interconnect several annotations or linguistic tools (like in a pipeline), a whole lot of conflicts and problems arise. In most cases, these conflicts and problems are so particular that they have to be solved *ad hoc* and on the fly.

1.2. THE PROBLEM OF LINGUISTIC TOOLS AND/OR ANNOTATIONS INTEROPERATION IN DETAIL

More specifically, the conflicts and problems that prevent linguistic tools and linguistic annotations from interoperating usually come from the enormous variation in the factors that are taken into

¹ http://semeval2.fbk.eu/semeval2.php?location=Rankings/ranking_task1.html.

account when designing and developing them. These factors can be technological (*i.e.*, computational) or theoretical (*i.e.*, linguistic).

Firstly, the main technological factors that hinder linguistic tool and linguistic annotation interoperation are the following:

1. The set of the possible input formats that each tool can take is constrained and most reduced. In several cases, linguistic tools admit only one type of input: a pure text file, an HTML file, a file with a text annotated following a peculiar schema and using an *ad hoc*-created tagset² and/or format, etc.
2. The output of each particular tool usually conforms to a particular and *ad hoc*-created schema as well. In most cases, these output schemas, their tagsets and their formats and/or structure do not comply with any kind of standard or guideline.

Secondly, there are also a number of **theoretical factors that prevent linguistic tools and linguistic annotations from being interoperable**. To begin with, when two linguistic tools have to interoperate, the sets of linguistic phenomena with which they deal do not often coincide; in fact, they are usually disjoint. For example, a natural language parser will focus on the syntax-related phenomena (or properties) of its input text, whilst a semantic tagger will focus on the sense of each lexical item in this same input text.

Furthermore, even when the same linguistic phenomenon is analysed or annotated by different but similar tools, the result depends, in most cases, on the linguistic model or theory underlying them. For example, (i) a natural language parser developed according to a functional perspective of human language will try to characterise the role (*i.e.*, the function) played by each input element within its context, whereas (ii) a natural language parser developed according to a structuralist perspective will try to determine, first of all, the constitution relations holding between these elements.

Consequently, different linguistic tools analysing or annotating the same input might yield disparate –and, still, correct– results, because (1) they only pay attention to a certain set of linguistic phenomena (that is, a certain set of morphological, syntactic, semantic, etc. properties of a certain language), and these sets of linguistic phenomena might not coincide or even be disjoint; and/or (2) they interpret this input according to a different perspective or linguistic theory. This is also manifested in their outputs, as with technological factors, by their tagset, format and structure.

² A **tagset** is the group of symbols (*i.e.*, **tags**) that are assigned and/or attached to the different elements of an input file in order to encode their resulting annotations or analyses.

Thus, briefly, the main theoretical (or linguistic) issues that can cause conflicts and problems when trying to make linguistic tools and linguistic annotations interoperate are

- A. The set of phenomena that each tool annotates and/or processes.**
- B. The type and number of the linguistic categories used by each tool to characterise a given set of phenomena or, in other words, the tagset designed to represent its corresponding analyses and/or annotations.** These tagsets differ from each other (1) when they focus on different aspects of the same phenomenon or, equivalently, when this phenomenon is interpreted according to different linguistic theories; (2) when the precision of the tags used to characterise it is not similar; or (3) when the meaning of the tags describing a given phenomenon are similar but the tags themselves are dissimilar (that is, they have been termed or encoded differently).
 - B.1. An instance of the first case is included in Table 1 and in Table 2 (see next page). These two tables show the analyses of the English sentence “John gave Mary an apple” outputted by Connexor’s Machine Phrase Tagger³ (a phrase structure syntactic parser) and Connexor’s Machine Syntax⁴ (a functional dependency parser), respectively. As shown in these tables, even though the input text is the same for both and the linguistic phenomena being analysed are similar (*i.e.*, the syntactic properties of the input text) the output is very different, though not completely disjoint. As for the analysis of the first word, ‘John’,
 - B.1.1 both outputs give its lemma or baseform (‘John’ in Table 1 and ‘john’ in Table 2) but they do not coincide, that is, it is performed differently or according to different criteria and/or theories;
 - B.1.2 whereas the first tool outputs the POS tag for ‘John’ (proper noun), its role in its phrase (nominal head) and the type of phrase it constitutes (single-word noun phrase), the second tool (i) outputs that ‘John’ has a subject syntactic dependency (subj:>2) on the second word of the sentence (*i.e.*, it is the subject of ‘gave’), (ii) restates that it is the subject of the sentence (@SUBJ), (iii) shows that it functions as a nominal head (%NH), and (iii) gives a less accurate POS tag for this word (N – noun) than the first tool, but supplemented by a partial morphological analysis (NOM – nominative case; SG – singular number). That is, whereas the first tool details phrase constituency relations and seems to yield more accurate POS tags, the second one focuses on syntactic dependencies and functions and gives more information about the morphological properties of words.
 - B.2. As for the second case, there are two main opposite trends when designing a tagset, namely
 - (a) including a minimal number of coarse-grained tags, which represent only the main categories needed to annotate each phenomenon and which allow for its easy and fast

³ <http://www.connexor.eu/technology/machine/demo/tagger/>

⁴ <http://www.connexor.eu/technology/machine/demo/syntax/>

annotation; and (b) including as many coarse and fine-grained tags as needed to further sub-characterise and unambiguously annotate the input elements. In the example mentioned in the previous paragraph, it has been already shown that the first tool yields a more fine-grained POS tag for ‘John’ (proper noun) than the second one (N – noun).

B.3. Lastly, there are also two main opposite trends to encode and/or to format the tags in a tagset, that is, (1) abbreviating and/or compressing them as much as possible, as in the example in Table 2, or as recommended for morphosyntactic annotation by the EAGLES project⁵ (for instance, tagging ‘John’ as a noun by means of the abbreviation N); and (2) expanding them as much as possible, as in the example in Table 1 (for instance, tagging ‘John’ by means of the POS tag ‘proper noun’ as such). Clearly, whereas the resulting tagsets from the former trend are more machine-readable, the ones resulting from the latter are more human-readable.

Table 1: Machinese Phrase Tagger analysis of the English sentence ‘John gave Mary an apple’.

Text	Baseform	Phrase syntax and part-of-speech
John	John	nominal head, proper noun, single-word noun phrase
gave	give	main verb, indicative past
Mary	Mary	nominal head, proper noun, single-word noun phrase
an	an	premodifier, determiner
apple	apple	nominal head, noun, single-word noun phrase
.	.	sentence boundary

Table 2: Machinese Syntax analysis of the English sentence ‘John gave Mary an apple’.

Text	Baseform	Syntactic relation	Syntax and morphology
John	john	subj:>2	@SUBJ %NH N NOM SG
gave	give	main:>0	@+FMAINV %VA V PAST
Mary	mary	dat:>2	@I-OBJ %NH N NOM SG
an	an	det:>5	@DN> %>N DET SG
apple	apple	obj:>2	@OBJ %NH N NOM SG
.	.		
<s>	<s>		

Thus, when comparing two linguistic tools, in each of these technological factors and theoretical issues there can be (i) a complete agreement (or a full overlapping); (ii) a complete disagreement (or no overlapping); or (iii) an intermediate degree of disagreement (or a partial overlapping). This implies that an *ad hoc* study must be done for each set of tools that is integrated. The purpose of such a study will be to determine precisely, on the one hand, the degree of agreement or overlapping in each

⁵ <http://www.ilc.cnr.it/EAGLES96/pub/eagles/corpora/annotate.ps.gz>

of these factors and issues and, on the other hand, the mapping between their ways to refer to their overlapping elements.

1.3. SOLVING THE PROBLEM: AN OUTLINE OF OUR PROPOSAL

In particular, when trying to integrate linguistic annotation tools,

1. the degree of overlapping between the set of phenomena they deal with and between their respective tagsets;
 2. and the mappings between their overlapping tags
- must always be determined beforehand.

Besides, it is always necessary to choose a format and tagset for the outputs of the integrated system, such that they can represent conveniently the results of each and every one of the different tools being integrated. The chosen output format and tagset, as a rule, will be different from the format and/or the tagset of some of the integrated tools. Therefore, a formal and detailed specification must be provided for the way to translate the outputs and tagsets of each tool into the output format and tagset of the integrated system.

1.3.1. THE ROLE OF ONTOLOGIES

It is precisely at this point where ontologies come into play⁶. Several experiments have already shown the usefulness of ontologies to solve computer-related integration and interoperation problems⁷. Additionally, in this particular case, some of the linguistic tools that had to be integrated were ontology-based to some extent (for instance, EuroWordNet). Thus, from the very beginning, ontologies became a key factor in the specification of a solution to the problem addressed. Accordingly, a scenario was envisaged in which

- 1) the tagsets associated to the different linguistic tools that had to be integrated be formalised in ontologies of linguistic knowledge;
- 2) the results (*i.e.*, the tags) outputted by each of these tools be translated into (or mapped onto) the terms included in these ontologies;
- 3) the outputs of the integrated system be expressed using these ontological terms as well.

⁶ As for the present work, an **ontology** is 'a formal, explicit specification of a shared conceptualization' (Gruber, 1993; Borst, 1997).

⁷ http://www.itcon.org/data/works/att/2010_14.content.02376.pdf;
<http://interop-esa05.unige.ch/INTEROP/Proceedings/Interop-ESAScientific/PerPaper/I05-1%20400.pdf>;
<http://mtg.upf.edu/files/publications/Troncy-Mareso-2007.pdf>;
<http://alexandria.tue.nl/repository/books/642889.pdf>;
<http://bulletin-mif.unde.ro/docs/20092/10GMoise2.pdf>.

In addition, this scenario had to be as general as possible, so that the ontologies developed in this process be highly reusable and language- and tool-independent. This required, basically,

- a) identifying also the linguistic phenomena not covered by the linguistic tools that had to be integrated, but which are analysed or annotated by different (though similar) ones, developed for a different language or following a different linguistic theory;
- b) formalising these additional phenomena into the linguistic ontologies;
- c) identifying the terms (mainly concepts) not included yet in the ontologies that allow the linguistic phenomena already formalised to be integrated and linked;
- d) formalising these other elements in the linguistic ontologies as well.

1.3.2. THE ROLE OF STANDARDISATION

Introducing standards and standardisation into this process of ontology development and generalisation seemed most helpful too. Indeed, standards and standardisation are proving to be key items when solving integration and interoperability problems⁸. Hence, both *de facto* and official standards had to be taken into account (1) when the tagsets of the involved linguistic tools were formalised into the ontologies; and also (2) when these ontologies were supplemented with the key terms not formalised yet. As a side effect, this would reaffirm the generality, the reusability and the tool- and language-independence of the ontologies.

1.3.3. A DESIDERATUM: MINIMISE CASCADED ERRORS

Nevertheless, the appearance of another problem, namely cascaded errors, was predicted at this stage. Cascaded errors appear when linguistic tools are connected in a pipeline. For instance, when a word is incorrectly POS-tagged this usually involves the incorrect tagging of this word (and sometimes even of its surrounding words) in higher annotation levels, such as the syntactic and the semantic.

More specifically, the Spanish word ‘busca’, depending on its context, can be (1) a common noun, either masculine (meaning ‘pager’, ‘beeper’ or ‘bleeper’) or feminine (meaning ‘search’); or (2) a verbal form of the verb ‘buscar’ (meaning usually ‘to look for/up’ or ‘to seek’), that is, either its second person, singular, imperative form, or its third person, singular, present tense and indicative mood form. Each of these four possible analyses corresponds with a different meaning and, hence, also with a different semantic tagging. Therefore, annotating ‘busca’ with an incorrect POS tag implies that its corresponding semantic tag will be necessarily incorrect as well.

⁸ Also when integrating linguistic annotations (<http://www.anc.org/MASC/About.html>).

Obviously, this integration problem (*i.e.*, cascaded errors) could diminish outstandingly the efficiency of the integrated system, but it could not be solved just by using ontologies. For this reason, the solution proposed for the integration of the different linguistic tools had to minimise as much as possible the impact of cascaded errors in the output of the integrated system.

1.4. OUR PROPOSAL: MAIN PILLARS, CONTRIBUTIONS AND RESULTS

So, to summarize, these are the main pillars on which this work was built: the research group owned some linguistic annotation tools that, in some cases, had been purchased at a fairly high price. Accordingly, in order to recoup their cost, they had to be integrated so that their combined results could be used from then on. Apart from mapping their results to the ontology-based tagset and format of the integrated system, this required (1) to simply interlink their outputs when their results did not overlap; and (2) to combine somehow, in a common and unique annotation, those results of the tools that overlapped. In addition, in the latter case, since some of the tools were connected in a pipeline, this also required to minimise the POS-tagging errors as POS results were combined. This contributed also to the minimisation of the cascaded errors yielded by the integrated system.

This achievement helped prove that, at least in this case, the interoperation of several POS taggers together with an adequate combination of their results can yield a much lower POS tagging error rate than the POS tagging error rate of any integrated POS tagging tool. An immediate consequence of this is that it is possible to obtain a fairly reliable automatic POS tagging from a set of not so reliable POS tagging tools, provided that some constraints are met.

The former assertion might be the main contribution of the present dissertation. However, it contains also some other important contributions. For instance, it presents a new model that enables linguistic tool interoperation, that is, the integration and combination of their results. This new model, which will be referred to as OntoTag, consists of two main elements, namely an annotation architecture and an integrative annotation scheme. Both are based on

1. the formalisation in ontologies of both (1) the results of the linguistic tools integrated in the final system and (2) their corresponding *de facto* and official standards; this comprises the knowledge associated to morpho-syntactic and syntactic annotations and, to a lesser extent, also to semantic annotations;
2. the mapping of the tags included in the annotations outputted by each tool for a given text onto their corresponding terms of these ontologies, which results in a sort of hybrid (that is, linguistic and ontological) and standardised tagset;

3. the use of these hybrid tagsets in conjunction with Semantic Web standardised languages, such as XML, RDF(S) and OWL, in order to attach hybrid, standardised annotations to the input being processed (web pages, in this case).

Finally, this work also aimed at showing, by means of a specific application, the potential uses of this hybrid annotation model in the context of the Semantic Web. For this reason, a new application was built and integrated together with the other linguistic tools. The main inputs of this new application were (1) a domain ontology about cinema and entertainment, developed on purpose, and (2) the annotations provided by the rest of the linguistic tools integrated in the system. The goal of this application was twofold: on the one hand, it had to recognise, classify and annotate the named entities⁹ of the domain selected that were present in the input texts; on the other hand, it had to include the annotated named entities as instances of their corresponding concepts in the domain ontology, thus enriching it. As a consequence, it was shown that, at least in certain occasions, the Semantic Web and Computational Linguistics could work together and benefit from the advances of each other.

1.5. STRUCTURE OF THIS DISSERTATION

This dissertation is organised as follows: Chapter 2 states the purposes and goals of the present work. Chapter 3 presents an introduction to the State of the Art of Linguistic Annotation and of the annotations for the Semantic Web. Chapter 4 introduces both the abstract annotation scheme and the abstract architecture of the model aforementioned. Chapter 5 shows some concrete implementations of the annotation scheme (OntoTagger's annotation schemas), as well as of the abstract architecture of the model (the OntoTagger's annotation configuration). The main results yielded by these applications of the model are analysed and discussed in Chapter 6. Chapter 7 summarises the main conclusions derived from the development of the present model and its implementations. Chapter 8 includes some issues that require further research, identified during the elaboration of the present work, and Chapter 9 the acknowledgements. Finally, Chapter 10 and Chapter 11 contain, respectively, the references and some additional and clarifying appendixes.

⁹ For a definition of named entity, see the State of the Art, page 33.

2. WORK OBJECTIVES

The present chapter details (a) the goals that had to be addressed in order to attain the main aim of this work (Section 2.1); (b) its potential contributions to the Semantic Web and the Corpus Linguistics (Section 2.3); and (c) the assumptions, hypotheses and restrictions applied within its development (Section 2.4).

2.1. OPEN RESEARCH PROBLEMS

As mentioned in the Introduction, linguistic annotation tools have still some limitations, which can be summarised as follows:

1. They only perform annotations at (a) certain linguistic level(s) (that is, Morphology, Syntax, Semantics, etc.).
2. They usually introduce a certain rate of errors and ambiguities when tagging. This error rate ranges from 10 percent up to 50 percent of the units annotated for unrestricted, general texts.
3. Their annotations are most frequently formulated in terms of an annotation schema designed and implemented *ad hoc*.

A priori, it seems that the interoperation and the integration of several linguistic tools into an appropriate software architecture could most likely solve the limitations stated in (1). Besides, integrating several linguistic annotation tools and making them interoperate could also minimise the limitation stated in (2). Nevertheless, in the latter case, all these tools should produce annotations for a common level, which would have to be combined in order to correct their corresponding errors and inaccuracies. Yet, the limitation stated in (3) prevents both types of integration and interoperation from being easily achieved.

In addition, most high-level annotation tools rely on other lower-level annotation tools and their output annotations to generate their own annotations. For example, sense-tagging tools (operating at the semantic level) often use POS taggers (operating at a lower level, *i.e.*, the morphosyntactic) to identify the grammatical category of the word or lexical unit they are annotating. Accordingly, if a faulty or inaccurate lower-level annotation tool is to be used by other high-level annotation tool in its process, the errors and inaccuracies introduced by the former in its annotations should be minimised in advance. Otherwise, these errors and inaccuracies would be transferred to (and even magnified in) the annotations of the high-level annotation tool.

Therefore, it would be quite useful to find a way to

- (i) correct or, at least, reduce the errors and the inaccuracies of lower-level linguistic tools;
- (ii) unify the annotation schemas of different linguistic annotation tools or, more generally speaking, make these tools interoperate.

Should the aforementioned problems be solved, the way would be cleared for annotating automatically web pages by means of linguistic tools, and transforming them into Semantic Web pages. Still, as mentioned above, one of the possible solutions to problem (ii) could be viewed as an interoperability problem as well. There again, ontologies have been successfully applied thus far to solve several interoperability problems. Hence, it seems that ontologies could also help solve the problems and limitations of linguistic annotation tools aforementioned.

Thus, to summarise, the main aim of the present work was to combine somehow these separated approaches, mechanisms and tools for annotation from Linguistics and Ontological Engineering (and the Semantic Web) in a sort of hybrid (linguistic and ontological) annotation model, suitable for both areas. This hybrid (semantic) annotation model should (i) benefit from the advances, models, techniques, mechanisms, tools and advances of both of these two areas; (ii) minimise (and even solve, when possible) some of the problems found in each of them; and (iii) be suitable for the Semantic Web. The concrete goals that helped attain this aim are presented in the following subsection.

2.2. GOALS OF THE PRESENT WORK

As stated above, the main goal of this work was to specify a hybrid (that is, linguistically-motivated and ontology-based) model of annotation suitable for the Semantic Web (*i.e.*, it had to produce a semantic annotation of web page contents – see Section 3.2.2.2). This entailed that the tags included in the annotations of the model had to (1) represent linguistic concepts (or linguistic categories, as they are termed in ISO/DCR (2008)), in order for this model to be linguistically-motivated; (2) be ontological terms (*i.e.*, use an ontological vocabulary), in order for the model to be ontology-based; and (3) be structured (linked) as a collection of ontology-based <Subject, Predicate, Object> triples, as in the usual Semantic Web languages (namely RDF(S) and OWL), in order for the model to be considered suitable for the Semantic Web.

Besides, to be useful for the Semantic Web, this model should provide a way to automate the annotation of web pages. As for the present work, this requirement involved reusing the linguistic annotation tools purchased by the OEG research group, but solving (or, at least, minimising beforehand) some of their limitations. Therefore, this model had to minimise these limitations by means of the integration of several linguistic annotation tools into a common architecture. Since this integration required the interoperation of tools and their annotations, ontologies were proposed as the main technological component to make them effectively interoperate. Consequently, it seemed evident

that a great step forward towards the formulation of such a model (henceforth referred to as OntoTag) would be the formalisation of the elements and the knowledge underlying linguistic annotations within an appropriate set of ontologies.

Obviously, first, to combine the results of the linguistic annotation tools that operated at the same level, their annotation schemas¹⁰ had to be unified (or, preferably, standardised) in advance. Second, to merge the results of the linguistic annotation tools operating at different levels, their respective annotation schemas had to be (a) made interoperable and (b) integrated¹¹. And third, in order for the resulting annotations to suit the Semantic Web, they had to be specified by means of an ontology-based vocabulary, and structured by means of ontology-based <Subject, Predicate, Object> triples, as hinted above. Therefore, a new annotation scheme had to be devised, based both on ontologies and on this type of triples, which allowed for the combination and the integration of the annotations of any set of linguistic annotation tools. This annotation scheme was considered a fundamental part of the model proposed here, and its development was, accordingly, another major objective of the present work.

All these goals, aims and objectives could be re-stated more clearly as follows:

Goal 1: Development of a set of ontologies for the formalisation of the linguistic knowledge relating linguistic annotation.

Sub-goal 1.1: Ontological formalisation of the EAGLES (1996a; 1996b) *de facto* standards for morphosyntactic and syntactic annotation, in a way that helps respect the <Unit, Attribute, Value> triple structure recommended for annotations in these works (which is isomorphic to the <Subject, Predicate, Object> triple structures used in the context of the Semantic Web).

Sub-goal 1.2: Incorporation into this preliminary ontological formalisation of other existing standards and standard proposals relating the levels mentioned above, such as those currently under development within ISO/TC 37¹² (see Chapter 3 – State of the Art).

¹⁰ That is, their tags (both their representation and their meaning), and their format or syntax.

¹¹ As for the present work, the verbs combine, integrate and merge, as well as their corresponding nominalizations (combination, integration and merging) are not considered synonyms when associated to annotations and will not be treated as such from now on. Combining annotations implies that the involved annotations belong to the same level and overlap to some extent. On the contrary, integrating annotations implies that the involved annotations belong to different annotation levels and/or do not overlap at all. The verb merge will be used as a hypernym of combine and integrate and, most frequently, to refer to a complex process that requires both combining and integrating annotations.

¹² The ISO Technical Committee dealing with Terminology (which also deals with linguistic resources and annotations).

Sub-goal 1.3: Generalisation and extension of the recommendations in EAGLES (1996a; 1996b) and ISO/TC 37 to the semantic level, for which no ISO/TC 37 standard has been developed yet.

Sub-goal 1.4: Ontological formalisation of the generalisations and/or extensions obtained in the previous sub-goal as generalisations and/or extensions of the corresponding ontology (or ontologies).

Sub-goal 1.5: Ontological formalisation of the knowledge required to link, combine and unite the knowledge represented in the previously developed ontology (or ontologies).

Goal 2: Development of a standard-based abstract scheme¹³ for the hybrid (linguistically-motivated and ontological-based) annotation of texts. This abstract scheme will be referred henceforth as OntoTag's annotation scheme.

Sub-goal 2.1: Development of the standard-based morphosyntactic annotation level of OntoTag's scheme. This level should include, and possibly extend, the recommendations of EAGLES (1996a) and also the recommendations included in the ISO/MAF (2008) standard draft.

Sub-goal 2.2: Development of the standard-based syntactic annotation level of the hybrid abstract scheme. This level should include, and possibly extend, the recommendations of EAGLES (1996b) and the ISO/SynAF (2010) standard draft.

Sub-goal 2.3: Development of the standard-based semantic annotation level of OntoTag's (abstract) scheme.

Sub-goal 2.4: Development of the mechanisms for a convenient integration of the three annotation levels already mentioned. These mechanisms should take into account the recommendations included in the ISO/LAF (2009) standard draft.

Goal 3: Design of an abstract architecture for the hybrid (semantic) annotation of texts (i) that allows for the integration and interoperation of different linguistic annotation tools, and (ii) whose results comply with the recommendations formulated in the abstract scheme of the previous goal. This abstract architecture will be referred henceforth as OntoTag's (abstract) annotation architecture.

Sub-goal 3.1: Specification of the decanting processes that allow for the classification and separation, according to their corresponding levels, of the results of the linguistic tools annotating at several different levels.

Sub-goal 3.2: Specification of the standardisation processes that allow (i) complying with the standardisation requirements of the abstract scheme developed within Goal 2, as

¹³ From now on, the term schema will refer to a concrete implementation of a scheme, which (1) is more abstract than a schema and (2) can be regarded as the specification of several possible schemas that differ, for example, in their implementation languages.

well as (ii) combining the results of those linguistic tools that share some level of annotation.

Sub-goal 3.3: Specification of the merging processes that allow for the combination of the output annotations and the interoperation of those linguistic tools that share some level of annotation.

Sub-goal 3.4: Specification of the merging processes that allow for the integration of the results and the interoperation of those tools performing their annotations at different levels.

Goal 4: Generation of a concrete instance of OntoTag's abstract scheme (henceforth referred to as OntoTagger's schema) for a concrete set of linguistic annotations, namely those performed by the tools and with the resources available in the research group (Bitext's DataLexica¹⁴, LACELL's (POS) tagger¹⁵, Connexor's FDG¹⁶, and EuroWordNet). This schema should help evaluate OntoTag against its main goals. Consequently, it should implement, at least, those levels of the abstract scheme dealing with the annotations of the set of tools considered in this implementation. This included the morphosyntactic, the syntactic and the semantic levels.

Goal 5: Implementation of a concrete instance of OntoTag's abstract architecture (henceforth referred to as OntoTagger's configuration), for this set of linguistic tools and annotations, and using the schema generated in the previous goal. This configuration should help support or refute the hypotheses of this thesis, stated in Section 2.4.2.

Sub-goal 5.1: Implementation of the decanting processes that allow for the classification and separation of the results of those linguistic resources that provide annotations at several different levels (on the one hand, LACELL's tagger operates at the morphosyntactic level and, minimally, also at the semantic level; on the other hand, FDG operates at the morphosyntactic and the syntactic levels and, minimally, at the semantic level as well).

Sub-goal 5.2: Implementation of the standardisation processes that allow (i) specifying the results of those linguistic tools that share some level of annotation according to the requirements of the schema developed within Goal 4, as well as (ii) combining these shared level results. In particular, all the tools selected perform morphosyntactic annotations and they had to be conveniently combined by means of these processes.

Sub-goal 5.3: Implementation of the merging processes that allow for the combination (and possibly the improvement) of the annotations and the interoperation of the tools

¹⁴ <http://www.bitext.com/EN/datalexica.asp>.

¹⁵ <http://www.um.es/grupos/grupo-lacell/quees.php>.

¹⁶ <http://www.connexor.eu/technology/machinese/glossary/fdg/>.

that share some level of annotation (in particular, those relating the morphosyntactic level, as in the previous sub-goal).

Sub-goal 5.4: Implementation of the merging processes that allow for the integration of the different standardised and combined annotations aforementioned, relating all the levels considered.

Sub-goal 5.5: Improvement of the semantic level of this configuration by adding a named entity recognition, (sub-)classification and annotation subsystem, which also uses the named entities annotated to populate a domain ontology, in order to provide a concrete application of the present work in the two areas involved (the Semantic Web and Corpus Linguistics).

2.3. EXPECTED CONTRIBUTIONS AND RESULTS

The expected contributions and results of the present work have been summarised below. They relate the areas of the Semantic Web and Linguistic Annotation and should follow the achievement of the goals presented in the previous section. They have been grouped for their presentation according to the goals that they relate to, that is, (i) the design and implementation of OntoTag's annotation architecture and (ii) the design and implementation of OntoTag's annotation scheme¹⁷.

As for the expected contributions coming from the design and implementation of OntoTag's annotation architecture, they can be summarised as follows.

First, interoperability is one of the hot topics not only for the linguistic annotation community but also in the whole Computer Science field. The specification (and implementation) of OntoTag's architecture for the combination and integration of linguistic (annotation) tools and annotations by means of ontologies should show at least one alternative way to make these different linguistic annotation tools and annotations interoperate in practice.

Second, as commented in the following section, the combination of the results of tools annotating at the same level was expected to yield better results (both in precision and in recall) than each tool separately. At least for the morphosyntactic level, this could be regarded as a way of constructing more robust and more accurate POS tagging systems.

And third, Semantic Web annotations are usually performed by humans or else by machine learning systems. Both of them leave much to be desired: the former, with respect to their annotation rate; the latter, with respect to their (average) recall. Hence, if linguistic tools could be wrapped in

¹⁷ Whether these expected contributions and results were eventually obtained or not is discussed in Chapter 7 (Conclusions).

order to automatically annotate Semantic Web pages using ontologies, this would entail their fast, robust and accurate semantic annotation.

As for the expected contributions resulting from the design and implementation of OntoTag's annotation scheme, they can be summarised as follows.

First, as shown in the State of the Art, there are different approaches and models for the semantic annotation of texts, but all of them focus on a particular view of the semantic level. Clearly, all these approaches and models should be integrated in order to bear a coherent and joint semantic annotation level. It was expected that OntoTag could help show (i) how these semantic annotation layers could be integrated together; and (ii) how they could be integrated with the annotations associated to other annotation levels.

Second, it was expected to obtain concrete examples on how standardisation (via ontologies, in this case) enables the combination, integration and interoperation of different linguistic tools and their annotations into a multilayered (or multileveled) linguistic annotation, which is one of the hot topics in the area of Linguistic Annotation.

And last but not least, should a way be found to coherently and uniformly formalise and annotate the different units and features associated to the different levels and layers of linguistic annotation, a great step would have been taken towards the global standardisation of this area, which is the aim of ISO/TC 37 (in particular, Subcommittee 4, dealing with the standardisation of linguistic annotations and resources).

Once the main (expected) contributions to the areas of the Semantic Web and Linguistic Annotation have been presented, it is the time to introduce the assumptions, hypotheses and restrictions underlying the development of the OntoTag model. This will be done in the following section.

2.4. WORK ASSUMPTIONS, HYPOTHESES AND RESTRICTIONS

This subsection discusses the main assumptions, hypotheses and restrictions underlying the development of the present work. First, the main assumptions taken into account as OntoTag was being devised are presented in Subsection 2.4.1; then, Subsection 2.4.2 shows the different hypotheses underlying this model. Finally, the different restrictions that affect the present work are discussed in Subsection 2.4.3.

2.4.1. ASSUMPTIONS

The main assumptions underlying the OntoTag model are presented in this section. They might affect and even determine to some extent the goals, and/or the scope of the present work.

First of all, it has been assumed that the final aim of the Semantic Web is to help humans, and to relieve them of the burden of processing the overwhelming amounts of information being generated and communicated nowadays. Hence, it makes no sense either to burden humans with the task of annotating one web page after another for computers to understand them all, as it is usually done in the context of the Semantic Web. Thus, the ultimate aim is to develop a most automatic process (1) to annotate and make explicit the content of Semantic Web pages, and then, (2) to process this explicit content.

However, there is much more meaning in Semantic Web pages than the meaning that is currently being made explicit by means of its kind of semantic annotations. Following Wierzbicka (1988), ‘all linguistic levels interact closely in order to determine the meaning of a whole sentence, utterance or expression’. Therefore, in order to make explicit as much content (*i.e.*, meaning) of web pages as possible, they should be annotated jointly at as many linguistic levels as possible. Besides, performing any kind of semantic annotation (even an ontologically-based one) usually requires some kind of morpho-syntactic and/or syntactic analysis of the web page being annotated. Not reusing and integrating the linguistic tools already developed for this purpose would be re-inventing the wheel as well as an unnecessary waste of time and efforts (and, also, maybe of money). This is another assumption underlying the present work.

Concerning the linguistic annotation tools, the following assumptions might have affected the design and the corresponding implementation of OntoTag’s abstract architecture:

- A.1. Linguistic annotation tools always accept an unformatted and untagged text (that is, clean or pure text) as input. On the contrary, most of them cannot tag HTML pages as such. However, the inputs of the model are, precisely, HTML (web) pages. Therefore, an optional distillation phase of the input files is assumed to be required. This phase will be responsible for removing all the HTML labels from the input files and extracting the clean text for its posterior annotation by means of all the tools involved.
- A.2. The annotations of the linguistic tools integrated in a concrete configuration of OntoTag may overlap or be ambiguous (either intentionally or not) at some level or layer. This is not a restriction of the model; instead, it is one of its requirements, in order to be as general as possible. Accordingly, when several tools are integrated into the same OntoTag’s architecture configuration, multiple alternative (or ambiguous) annotations might be attached to the linguistic

units in the input at a given level or layer. However, the annotation assigned by OntoTag implementations to each linguistic unit in the input should be unique (whenever possible) for a given level and layer. This entails the assumption that a combination sub-phase must be included in the architecture (configurations) of the model. This combination sub-phase must output a unique annotation for each of the linguistic units found in the input text when two or more annotations overlap (or are ambiguous). How this unique annotation is obtained was one of the main problems to be solved in the present work (see Section 4.2.5).

- A.3. Some linguistic tools perform more than one type of annotation on the input text (that is, pertaining to different levels and/or layers). As a consequence, in order for the model to be general enough, it must assume that some of the tools incorporated into its architecture configurations might tag the input documents at more than one level. However, the model also assumes that the annotations of these tools might overlap at some level or layer and that they need to be combined according to this level and/or layer. As for the model configurations, this yields the need for both (i) a decanting phase before the same-layer (or same-level) annotations can be combined and (ii) an integration sub-phase after they have been conveniently combined. Whilst the decanting phase will separate the annotations coming from each tool according to their level, the integration sub-phase will effectively put together the combined annotations of all the levels and layers considered.
- A.4. Finally, linguistic annotation tools have been developed according to some particular annotation commitments or theories and their annotation schemas do not usually conform to the existing (*de facto*) standards or standard proposals for linguistic annotation. Consequently, it must be assumed that, in general, the annotation schemas and the tagsets of the different linguistic tools integrated into an OntoTag's configuration will not match. Hence, their annotations must be mapped in advance according to a common annotation scheme(a) and a common tagset. This is assumed to enable and ease their automatic comparison and combination. From both this assumption and from Assumption A.2 arises the need for a standardisation phase in OntoTag's architecture.

2.4.2. HYPOTHESES

The model developed in the present thesis tries to shed some light on (i) whether linguistic annotation tools can effectively interoperate; (ii) whether their results can be combined and integrated; and, if they can, (iii) how they can, respectively, interoperate and be combined and integrated. Accordingly, several hypotheses had to be supported (or rejected) by the development of the OntoTag model and its implementation. These hypotheses are the following:

- H.1. The annotations of different levels (or layers) can be integrated into a sort of overall, comprehensive, multilayer and multilevel annotation, in order for their elements to complement and refer to each other.
- H.2. Tool-dependent annotations can be mapped¹⁸ onto a sort of tool-independent annotations and, thus, be standardised.
- H.3. This type of standardisation should ease
 - H.3.a. the interoperation of linguistic tools.
 - H.3.b. the comparison, combination (at the same level and layer) and integration (at different levels or layers) of their respective annotations.
- H.4. Ontologies and Semantic Web technologies (can) play a crucial role in the standardisation of linguistic annotations, by providing consensual vocabularies and standardised formats for annotation (*e.g.*, RDF triples).
- H.5. The rate of errors introduced by a linguistic tool at a given level when annotating can be reduced automatically by contrasting and combining its results with the ones coming from other tools operating at the same level. However, these other tools might be built following a different technological (stochastic *vs.* rule-based, for example) or theoretical (dependency *vs.* HPS-grammar-based, for instance) approach.
- H.6. Each linguistic level can be managed and annotated independently.

2.4.3. RESTRICTIONS

The following restrictions must be taken into account when interpreting the results and the scope of both the OntoTag model and its corresponding implementations:

- R.1. The input documents of the model and its implementations are HTML web pages.
- R.2. The output documents of the model and its implementations have to be Semantic Web documents.
- R.3. Ontologies are required to be consensual. A wide consensus in Linguistics can be achieved by means of some projects and initiatives aiming at the standardisation of linguistic annotations. Therefore,
 - R.3.a. The more linguistic standards and standard proposals (such as those of ISO/TC 37) and results from standardisation projects and initiatives (such as EAGLES, SIMPLE and ISLE) that are taken into consideration when developing OntoTag's (linguistic) ontologies, the better.

¹⁸ In its mathematical sense, that is, make a correspondence.

- R.3.b. Sticking just to a particular linguistic grammar or theory in the development of OntoTag must be avoided.
- R.4. In particular, the abstract annotation scheme and its implementation should comply with
- R.4.a. The EAGLES (1996a, 1996b) recommendations regarding a layered and a `<Unit, Attribute, Value>` triple-based formulation of annotations.
- R.4.b. The ISO/TC 37/SC4 corresponding standards and standard proposals.
- R.5. In addition, the W3C has already deployed several standard languages for the Semantic Web, such as XML, RDF(S) and OWL. For this reason, one (or each) of them should be chosen for the implementation of the corresponding annotation schemas.
- R.6. The ontologies associated to the model should be stored and generated by means of the OEG's¹⁹ Ontology Development Environment, WebODE²⁰.
- R.7. One of the advantages of WebODE is that it also allows for a subsequent querying of the ontologies stored in it. This can be done by means of a Java API, developed for that purpose. Accordingly, for the sake of interoperability and compatibility, Java had to be the implementation language of the model configurations. This also offers an additional advantage over other programming languages: the potential independence from the operating system of the application being developed.
- R.8. Also for the sake of interoperability and compatibility, the operating system under which this configuration should be developed, tested, installed and run is also fixed (to Windows 2000 Professional) by another policy of the research group (OEG).
- R.9. Spanish is the language chosen for the input files to be annotated in order to comply with the objectives of the first project that partly funded the present research, namely ContentWeb²¹.
- R.10. For the same reason, the domain of application of all the experiments carried out in the context of the present work has been set in advance to the entertainment domain.
- R.11. At the moment of elaborating the goals of the present work, there were no Spanish-related freely available tools that could be used for the implementation of the abstract architecture of OntoTag²². Therefore, the only linguistic tools and ontologies available for this implementation were the ones already existing in the OEG, that is, the aforementioned tagging tools (Bitext's DataLexica and Connexor's FDG) and EuroWordNet. However, the LACELL research group of the Universidad de Murcia kindly provided the group with the morphosyntactic annotation of a sample of the ODECORPUS-Entertainment (consisting of around 5000 words), which was incorporated into the present study as well.

¹⁹ <http://www.oeg-upm.net>.

²⁰ <http://webode.dia.fi.upm.es/WebODEWeb/index.html>.

²¹ See also <http://www.oeg-upm.net>.

²² The development of OntoTag started in 2002. Some linguistic annotation tools for Spanish are currently freely-available, such as FreeLing (<http://www.lsi.upc.edu/~nlp/freeling/>).

- R.12. For this reason, the evaluation of the hypothesis concerning the combination and the integration of annotations was restricted to morphosyntactic, syntactic and semantic annotations, which are the ones provided by these tools.
- R.13. However, the range of linguistic tools available only allowed for the experimentation and evaluation of the combination sub-phases of the model dealing morphosyntactic annotations (*i.e.*, lemma combination, morphosyntactic category combination and morphological [attribute-value] combination). These were the phases for which the annotations obtained from these linguistic tools overlapped.
- R.14. Finally, due to the sort of input documents expected (mostly web pages), and also for the sake of time and human resources, those linguistic levels dealing with oral corpora (such as the phonetic or the prosodic) have been neglected within the present work.

This concludes the specification of the goals, the expected contributions, the assumptions, the hypothesis and the restrictions of the present work. Chapter 4 (page 71) and Chapter 5 (page 209) discuss, respectively, the OntoTag model and its first prototype (OntoTagger), developed in order to attain the aforementioned goals. The next chapter presents the state of the art of the technologies, annotations and standards involved in their development.

3. STATE OF THE ART

As explained in the previous chapters, the main aim of the present work was to devise a hybrid (*i.e.*, linguistic and ontological) annotation model (i) that benefited from the advances in both Linguistics and Artificial Intelligence (or, more concretely, in Ontological Engineering and the Semantic Web) and (ii) that was suitable and useful for the Semantic Web and its purposes.

Thus, a survey of the State of the Art on annotation had to be elaborated in order to (i) identify and introduce the terminology used in these two areas to refer to the different elements concerning annotation; (ii) find out the technologies, models, languages, mechanisms, devices and all other resources applied to annotation in each of them; (iii) determine the similarities and complementarities of these resources and the way to combine them and make them interoperate; (iv) establish which additional resources needed to be developed to make them effectively interoperate. The resulting survey has been summarised in this chapter.

Accordingly, this chapter summarises the State of the Art of annotation when the present thesis started being developed in both Linguistics and Ontological Engineering (and the Semantic Web). In other words, in this chapter it is presented what, how and why was annotated when the present annotation model was devised. However, some aspects have been updated as the present dissertation was being written, in order to show that (i) its preliminary results, when published, were original and important contributions to the State of the Art at that time and (ii) its main results, not published yet, are still original and might constitute additional contributions to the current State of the Art.

Even though annotation is a hot and most fashionable topic nowadays in both Linguistics and Web-related forums, it is almost as ancient as writing and written texts, that is, as History. Therefore, most of the advantages and problems of annotations identified hitherto were already present in ancient annotations. Accordingly, the solutions and conventions adopted for some of them as well as some properties of annotations can be applied nowadays and/or are still on debate.

For this reason, two selected historical forms of annotation are summarised in the following section (Section 3.1), so as to (i) introduce the current basic terminology in the annotation domain and (ii) illustrate its commonplace elements. Then, the main current approaches to annotation (*i.e.*, the linguistic and the computational) are summarised in Section 3.2, making use of the terminology and the elements introduced in Section 3.1, extending and detailing them when necessary according to each approach needs and purposes. Finally, Section 3.3 mentions the main overlaps and complementarities of these two approaches in order to build a hybrid (that is, linguistic and ontological) annotation model, which were taken into account when carrying out the present work.

3.1. ANNOTATION: A HISTORICAL APPROACH AND BASIC TERMINOLOGY

According to OALD (2005), ‘to annotate’ is ‘to add notes to a book or text, giving explanations or comments’, a ‘note’ is ‘a short comment on a word or passage in a book’. So, briefly, **annotation** could be defined as *the action and/or the effect of adding (short) comments or explanations to a word, a text or a book*²³.

From this point of view, annotation dates as back as Ancient Egypt and pharaohs times, that is, more than 2000 years ago. In those times, Egyptian words were encoded in Egyptian writings (both manuscripts and inscriptions) by means of one or more hieroglyphs, which described how the word sounded. In addition, a small subset of Egyptian hieroglyphs was used to add comments or explanations to those associated to some Egyptian (written) words. The hieroglyphs included in this small subset are called determinatives nowadays. Whereas the hieroglyphs used to write words were associated a sound when read, determinatives were not. **Determinatives** stood immediately after the hieroglyph(s) associated to a word and were used mainly to explain the concrete *sense* (or meaning) of the hieroglyph or set of hieroglyphs (*i.e.*, the word) after which they were included. A list of the most commonly used determinatives is shown in Figure 1.

For instance, in written ancient Egyptian, there were polysemous (that is, ambiguous) words, like ‘check’ in English, which might mean (i) ‘bank check’ (or cheque), (ii) a particular configuration of chess pieces or ‘verification’²⁴. More concretely, in ancient Egyptian, the sequence of hieroglyphs ^J (standing for the sounds ‘w’ and ‘n’, respectively, and which is pronounced /wen/) was polysemous, and required a proper disambiguation. Its possible meanings were (i) ‘open’, (ii) ‘hurry’, (iii) ‘mistake’, (iv) ‘become bald’, (v) ‘Hermopolis’ and (vi) ‘light’ (Zauzich, 1992;2004). Consequently, a corresponding determinative was added at the end of this sequence in order to disambiguate its meaning in a given context.

Thus, (i) a door (shown in Figure 1, lowest row, third column), (ii) a running legs (8), (iii) an ‘evil’ bird ({}), (iv) a lock of hair (shown in Figure 1, lowest row, first column), (v) a city with crossed roads (<) and (vi) a sun with rays determinative hieroglyph, respectively, was attached at the end of the polysemous sequence, *i.e.*, ^J. **Accordingly, as shown by this example, annotations have been or can be useful in disambiguating the sense (or the meaning) of words.**

²³ This definition is corroborated by the ones included in the Merriam-Webster Online Dictionary (<http://www.m-w.com/>), which gives two definitions of ‘annotation’: (a) a note added by way of comment or explanation and, (b) the act of adding notes. For a discussion on the different terms used to refer to the process and the results of annotation, such as **tag**, **label**, **markup**, **tagging** or **labelling**, see Aguado de Cea et al. (2009).

²⁴ As for a Spanish example, the word ‘banco’ has several different meanings or senses, such as ‘bench’, ‘bank’ or ‘shoal’.



























































































 man, person	 eye, see, actions of eye	 plant, flower	 box, coffin
 woman	 actions or conditions of the eye	 vine, fruit, garden	 shrine, mat
 people	 nose, smell, joy, contempt	 wood, tree	 boat, ship, navigation
 child, young	 ear, relating to the ear	 grain	 sacred bark
 old man, old	 tooth, actions of teeth	 sky, above	 clothe, linen
 official	 force, effort	 sun, light, time	 bind, document
 exalted person, the dead	 offer, present	 night, darkness	 rope, actions with rope or cord
 god, king	 arm, bend arm, cease	 star	 knife, cut
 king	 envelop, embrace	 fire, heat, cook	 hoe, cultivate, cut up
 god, king	 phallus, beget, urinate	 air, wind, sail	 break, divide, cross
 goddess, queen	 leg, foot, actions of foot	 stone	 cup
 high, rejoice	 walk, run	 copper, bronze	 vessel, anoint
 praise, supplicate	 move backwards	 sand, minerals, pellets	 pot, vessel, beverages
 force, effort	 limb, flesh	 water, liquid, actions connected with water	 bread, cake
 eat, drink, speak, think, feel	 tumors, ordors, disease	 sheet of water	 loaf, cake, offering
 lift, carry	 bodily discharges	 irrigated land	 festival
 weary, weak	 cattle	 land	 abstract, book, writing
 enemy, foreigner	 skin, mammal	 road, travel, position	 royal name, king
 enemy, death	 bird, insect	 desert, foreign country	 one, the object shown
 lie down, death, bury	 small, bad, weak	 foreign	 several, plural
 mummy, likeness, shape	 fish	 town, village, Egypt	 substitute for signs difficult to draw (mostly hieratic)
 head, nod, throttle	 snake, worm	 house, building	
 hair, mourn	 tree	 door, open	

Figure 1: Determinative hieroglyphs (© Copyright 1997, Jim Loy)²⁵²⁵ Taken from <http://www.jimloy.com/hiero/determin.htm>.

Besides, the first two symbols of the first column of the figure were used to indicate that the word (*i.e.*, the hieroglyphs) to which they were attached had a *masculine gender*, if the determinative attached was a (a seated man), and *feminine*, if the determinative attached was b, which depicts a seated woman (Donoughue, 1999;2002). The determinative in the last row of the last column (3), was added to indicate a *plural number*. An example of their (conjoint) application has been included in Figure 2, which shows how the corresponding *gender* and *number* of brother, sister, brothers and sisters were annotated by means of determinatives. Thus, as shown by this example, **annotations have been used and can be used to enumerate or make explicit the properties of the words or the text to which they are attached.**

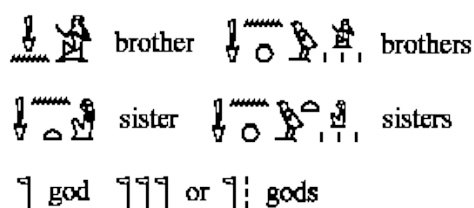


Figure 2: Use of determinative hieroglyphs to annotate gender and number (© Copyright 1997, Jim Loy)²⁶

Nevertheless, *number* and *gender* were realised in ancient Egyptian by means of their corresponding (sound) suffixes, which had a matching hieroglyph when written. On the one hand, a *feminine gender* was marked by a ‘-t’ suffix and depicted by a bread piece hieroglyph (V), whereas a *masculine gender* was marked by an empty suffix. On the other hand, a *plural number* was marked by a ‘-w’ suffix or a ‘-nw’ suffix and depicted by a quail chick hieroglyph (E) or by a spherical jar followed by a quail chick hieroglyph, respectively. A *singular number*, on the contrary, was marked by an empty suffix (Collier & Manley, 1998; Zauzich, 1992;2004). Subsequently, adding a determinative to indicate the *gender* or the *number* of a word was actually redundant. This is another characteristic of **annotations: they are or can seem (at least, to some extent or for some of its readers) redundant**, since they re-express somehow some information that is explicit or implicit in the text being annotated. **Therefore, it is important to place them where they are not an annoyance to a reader that does not need them to understand the text.** Moreover, whenever possible (and especially when most of the text readers might not need these annotations to understand the text itself), for the sake of readability, annotations should be kept in a separate location (*i.e.*, document), but conveniently linked to the parts of the text to which they are added. This strategy is known in the linguistic field as **stand-off annotation** (ISO/LAF, 2009; ISO/MAF, 2008). It clearly contrasts with the way in which these ancient annotations were added, since they were supposed to clarify the meaning of the text itself for most of its readers.

²⁶ Taken from <http://www.jimloy.com/hiero/plural.htm>.

Yet, there is another example of historical (and most significant, at least for Spanish and Basque) annotation: the *Glosas Emilianenses*²⁷. The *Glosas Emilianenses* (a Spanish expression for “glosses of [Saint] Emilianus”) are glosses²⁸ written in a Latin codex, originally located in a monastery of San Millán (≈ “Saint Emilianus”) de la Cogolla (La Rioja, Spain). These glosses were written a thousand years ago in three languages, namely (i) a simplified version of Latin, (ii) the medieval form of a Hispanic Romance language (traditionally regarded as Castilian or Old Spanish) and (iii) medieval Basque (just two glosses)²⁹. The aim of the author of these glosses was to make the text glossed more understandable to those readers not acquainted with its original language, *i.e.*, Latin. Therefore, he annotated either (i) full sentences or (ii) just isolated and uncommon (strange) words. The resulting glosses consisted of their translation into one more of the other three languages aforementioned, which the expected readers of the text were supposed to understand better. This is a most relevant use of **annotations** as for the present work: they **have been used and can be used to re-express the meaning of the words or the sentences of a text in a way that can be understood (better) by a particular type of targets or readers – be they humans or machines, as in (Semantic) Web annotations.**

Some other prominent characteristics and properties of annotations can be extracted from these historical examples. **First, two different types of information can be distinguished in annotated texts: (i) the content of the text and (ii) the annotations added to this content.** The former is the information included originally in the text, whilst the latter is the information added in order to explain, emphasise or make the former more readable. In the case of Egyptian writings, the determinatives constitute the annotation information added to the actual content of the text. In the case of the *Glosas Emilianenses*, the original text was the content, and the glosses were the additional information added to the text to explain this content.

Second, two different types of vocabularies can be differentiated in annotated texts, each one corresponding to the two different types of information that they carry. Accordingly, annotated texts contain words of both **a vocabulary that encodes its content and a vocabulary that encodes its annotations.** These two vocabularies may coincide or not, but they usually differ, in order to avoid ambiguities between the content and the annotations. In Egyptian writings, whereas the set of determinatives constitutes the vocabulary that encodes the annotations, the rest of hieroglyphs constitute the vocabulary used to encode the content of Egyptian texts. In the *Glosas Emilianenses*, the

²⁷ http://es.wikipedia.org/wiki/Glosas_Emilianenses

²⁸ A gloss is a ‘note or comment added to a piece of writing to explain a difficult word or phrase’ (OALD, 2005).

²⁹ The manuscript and these glosses are significant precisely for its early examples of writing in Basque and a form of Spanish. In fact, San Millán de la Cogolla has earned the reputation as the “birthplace of the Spanish language” because of these glosses, which were also important in its designation as a World Heritage Site (“cultural” type) in 1997 (cf. http://es.wikipedia.org/wiki/Glosas_Emilianenses; <http://whc.unesco.org/en/list/805>).

content uses a (standard) Latin vocabulary, whilst the glosses are expressed in an evolved form of Latin, a form of proto-Spanish or an ancient form of Basque.

Each elements of the vocabulary used to annotate the content of texts is usually referred to as a **tag**, a **label** or, simply, as an **annotation**. Accordingly, the whole vocabulary used to annotate somehow a set of texts referred to as its **tagset** or its **metadata**; and the way in which the tags of a tagset can be combined in order to express complex or higher-level annotations for text content will be referred to as an **annotation language** here. Clearly, as shown by these two ancient examples of annotations, both natural and artificial (or formal) languages and/or vocabularies can be used in an annotation language.

Third, also to avoid possible ambiguities between the content and the annotations, **they must be separated somehow in the annotated texts. This is usually achieved following a particular set of rules when annotating.** These rules fix and constrain (i) the vocabulary that can be used to encode the annotations, (ii) the place where they have to be added and/or (iii) the way in which they can be added. As for Egyptian manuscripts and inscriptions, the annotations had to be expressed just by means of determinatives, which were added at the end of (*i.e.*, after) the hieroglyphs that encoded the word that they commented. As for the *Glosas Emilianenses*, (i) they were added on the margins of the manuscript or between parentheses and (ii) they were encoded according to the rules and vocabulary of the three languages mentioned above, which differed from the original language of the text. A set of rules that has to be followed to separate content from annotations will be referred to as an **annotation scheme** (or **schema**)³⁰ here.

As explained below, the intended use and the targets of annotations, and also the way in which annotation languages and schemas separate content from annotations, are very important aspects in the design of this type of languages and schemas nowadays. This is shown more clearly in the following section, which presents the current forms and schemes (or schemas) of annotation in both the linguistic and the computational approaches.

3.2. ANNOTATION: CURRENT APPROACHES

Back to the present time, there are two main approaches to annotation nowadays, namely the linguistic and the computational. Each of them is presented and discussed in a dedicated subsection below, respectively, Subsection 3.2.1 and Subsection 3.2.2.

³⁰ The term **annotation scheme** will be used here to refer to a sort of specification of an **annotation schema**, which refers preferentially to a concrete implementation of an annotation scheme.

3.2.1. THE LINGUISTIC APPROACH TO ANNOTATION

At present, linguists working in annotation are interested in having a large sample of documents or text (a **corpus**) with its meaning described by means of some form of annotation; this annotation can be undertaken manually (that is, performed by **human annotators**) or automatically (that is, performed by or with **annotation tools**).

Automatic annotation can vary from computer-assisted to fully automatic. **Computer-assisted annotation** assigns candidate labels to all the words in a text leaving for manual treatment those words that the system does not know and also those which remain ambiguous even after applying disambiguation routines. In **fully automatic annotation**, the final aim is to assign the correct tags deterministically to all known linguistic units (or categories) in a text, without any manual intervention and without leaving any ambiguous unit.

The different kinds of annotations included in corpora have been traditionally referred to as **linguistic annotation levels**. However, the term **linguistic annotation layer** has been coined more recently to refer to the particular subtypes of annotations that can be distinguished within a given linguistic annotation level. Thus, a survey of the different linguistic annotation levels and their corresponding layers is presented next.

The linguistic annotation levels and layers considered here are the ones covered to a greater or to a lesser extent by the tools eventually used to evaluate the present thesis (discussed in Subsection 5.1, page 209). They are also the ones considered within the EAGLES (1996a; 1996b; 1999) recommendations. Thus, the annotation levels and layers introduced in the next subsections are the following: morphosyntactic annotation (that is, lemma and part-of-speech tagging), syntactic annotation and semantic annotation.

3.2.1.1 MORPHOSYNTACTIC ANNOTATIONS

Morphosyntactic annotations are the linguistic annotations that belong to the morphosyntactic level of annotation. There are two (main) types of morphosyntactic annotations, namely lemma and part-of-speech tagging. Each of them is described below.

On the one hand, **lemma tagging** accompanies every word-token in a text with its **lemma**, that is, the head word form that one would look up when looking for that word in a dictionary (for example, the lemma tag for the Spanish word ‘vayáis’ (≡ ‘[that you] go’) would be ‘ir’ (≡ ‘go’). As shown in the example, in English, *lemma tagging* may be considered redundant. However, in more highly-inflected languages, such as Spanish, the ratio of *word forms* per lemma makes *lemma tagging* a very valuable

contribution to information extraction (Leech, 1997). *Lemma tagging* is performed within the **lemma tagging layer** of the morphosyntactic level. The automatic tools that perform this type of annotations are usually called **lemma taggers** or *lemmatisers*, but these tools are scarcely developed separately. Instead, they are usually part of other more general morphosyntactic annotation tools (*i.e.*, *POS taggers*), described below.

On the other hand, **part-of-speech tagging** is one of the most extended types of linguistic annotation, together with the syntactic annotation. It is also referred to as *POS tagging* or *grammatical tagging*. It consists in the annotation of the grammatical class (*e.g.*, *noun*, *verb*, etc.) of each *word* or *token* in a text, together with (possibly) the annotation of its morphological attributes (*e.g.*, *gender* and *number*) and their corresponding values (such as *masculine*, *feminine*, etc., or *singular*, *plural*, etc., respectively).

Obviously, this type of annotations deals with two different types of elements, that is, grammatical categories (or classes) and morphological features³¹. Therefore, as for the present work, they are assumed to be performed each in a dedicated morphosyntactic annotation layer, namely the **morphosyntactic category annotation layer** and the **morphological annotation layer** (respectively).

Nevertheless, these two layers, together with the lemma tagging layer (as commented above), are implemented as different functions of **POS taggers**, which is the usual term to refer to morphosyntactic annotation tools. A computer can carry out this task currently fairly accurately without manual intervention, it must not be thought of as trivial. However, as was clearly shown by the experiments carried out in the development of the present work (see Chapter 6 –Results and Evaluation), in general, the error rate associated to POS taggers raises up to almost 15-20 percent of the words (or tokens) annotated.

However, POS information constitutes an essential foundation for further forms of annotation, such as syntactic and semantic annotation (McEnery & Wilson, 2001). Therefore, most likely, an error at the identification of the grammatical category or the lemma of a morphosyntactic category will entail a wrong syntactic and semantic annotation of this unit and also of its surrounding ones. Accordingly, in order for a hybrid (both linguistic and ontological) annotation of texts to be accurate, this error rate must be reduced as much as possible, which is one of the main goals of the present work.

The EAGLES (1996a) recommendations and the ISO Morphosyntactic Annotation Framework (ISO/MAF, 2008) are the most relevant **standardisation initiatives** dealing with morphosyntactic annotation.

³¹ A linguistic feature is a <Linguistic Attribute, Linguistic Value> pair.

On the one hand, the **EAGLES (1996a) recommendations for the morphosyntactic annotation of corpora** identify and list (i) the core set of morphosyntactic categories that *must* be annotated in order to minimally comply with these recommendations and (ii) the secondary categories and the main attributes and values that *should* or *could* be annotated at this level. In other words, EAGLES (1996a) recommendations consider *obligatory* the annotation of the major morphosyntactic categories or parts of speech (*noun, verb, adjective, adverb*, etc.), whereas the annotation of their subcategories and features (*i.e.*, attribute-value pairs) is considered either *recommended* or *optional*, depending on the relevance and the (number of) languages that include the subcategory or feature in question. Hence, (1) subcategories such as *common noun* or *proper noun* or attributes such as (1.a) *gender, number* or *case* for nouns, (1.b) *person, gender, number, tense, voice*, etc. for verbs, and (1.c) *degree, gender, number* and *case* for adjectives (together with their main values) are considered *recommended*; and (2) subcategories such as *countable noun* or *mass noun*, and the *aspect* attribute of verbs (together with its corresponding values) are considered *optional* in these recommendations.

However, these EAGLES (1996a) recommendations covered mostly the morphosyntactic categories (or units), attributes and values associated to the Western-European languages. Accordingly, these recommendations could be considered biased to some extent or from some point of view. The ISO/MAF (2008) standard tries to make up for this bias, that is to say, to cover other languages.

Thus, on the other hand, the **ISO Morphosyntactic Annotation Framework (ISO/MAF) standard** ‘tries to delimit minimal and maximal sequences in documents that can be identified as morphosyntactic units and tries to categorise the linguistic properties that may be used to mark these units, within some larger context’ (ISO/MAF, 2008). Therefore, the scope of ISO/MAF is evidently wider than the one assumed in the elaboration of the EAGLES (1996a) recommendations, whose units were (isolated) *words* and *punctuation marks*. Besides, this standard addresses and tries to solve common and traditional problems not contemplated in EAGLES (1996a) recommendations, such as

- (i) where to place the (morphosyntactic) annotations: (a) into the document being annotated (**embedded notations**) or (b) separated in a different one, with the annotations linked by references to their corresponding units in the original document (**stand-off notation**);
- (ii) how the different units in the original text follow each other: (a) **joint** as, for example, in the French expression ‘*L’on dit*’, or (b) **overlapping**, as, for instance, the French ‘*des*’ (= ‘*de les*’ ≈ ‘of the’) and Spanish ‘*del*’ (*id.*) contractions;
- (iii) how to annotate **multiword tokens**, such as ‘Prime Minister’, **discontinuous tokens**, such as the German separable verb ‘*aufsteigen*’ ≈ ‘(to) get up’ in the sentence ‘*Morgens steige ich um 8:00 Uhr auf*’ (≈ *‘In the morning, get I at 8:00 up’ ≈ ‘I get up at 8:00 o’clock in the

morning') , or **compound word forms**, such as the corresponding German and English word forms 'Geburtstag' and 'birthday'.

3.2.1.2 SYNTACTIC ANNOTATIONS

Once morphosyntactic units (or categories) in a text have been identified, *syntactic annotations* add information about the syntactic relationships holding between them. These syntactic relationships are determined, *e.g.*, by means of a phrase-structure or dependency parse. Different parsing schemes are employed by different annotators; according to McEnery & Wilson (2001) these schemes differ in

- The number of constituent types they employ (typically, the number of tags in the POS tagset).
- The way in which constituents are permitted to combine with one another.
- The grammar followed to parse and annotate the text.

The main **standardisation initiatives** dealing with syntactic annotation are (i) the EAGLES (1996b) recommendations and (ii) the ISO Syntactic Annotation Framework standard (ISO/SynAF, 2010).

As for the **EAGLES (1996b) recommendations for the syntactic annotation of corpora**, on the one hand, they detail the different layers of syntactic annotation identified within the syntactic annotation level, namely

- a) The **bracketing layer**, which aims at recognising and segmenting the different syntactic units (sentences, clauses, phrases and tokens) that constitute an expression.
- b) The **syntactic category annotation layer**, whose goal is to indicate the formal category of the syntactic units identified in the bracketing layer, such as *noun phrase*, *verb phrase*, *relative clause*, etc.
- c) The **syntactic dependency relation labelling layer**, which seeks to show the relations that hold between the category that functions as the head of a higher rank³² syntactic category and the other syntactic categories that constitute the higher rank syntactic category (*i.e.*, which depend of the head aforementioned). For example, it aims at showing the relation that holds between an *adjective* and the *noun* that it modifies in a *noun phrase*. This type of annotation is usually restricted to words (tokens).
- d) The **syntactic function labelling layer**, that is, the layer in charge of tagging the constituents of a higher rank syntactic category (or unit) according to the syntactic function they perform in it, such as *subject*, *object*, *adjunct*, etc.

³² See layer (g).

- e) The **syntactic feature annotation layer**, whose aim is to assign their corresponding features to the syntactic category identified in the *syntactic dependency relation labelling layer*, e.g., to annotate a *noun phrase* with a *singular number*, or a *verb phrase* with a *past tense*.
- f) The **logical relation labelling layer**, which entails the annotation of a variety of syntactic phenomena, such as *ellipsis*, *control*, *traces* and *syntactic discontinuity*.
- g) The **rank annotation layer**, which should make explicit the constituency relations that hold between the different syntactic units in an expression, in case it is not marked explicitly, although it could be obtained from the embedding of the brackets annotated in the bracketing layer.
- h) The **non-fluency annotation layer**, which is responsible for showing a range of **phenomena** that usually occur in **spoken language** transcriptions and corpora (e.g., *blends*, *false starts*, *reiterations* and *filled pauses*).

On the other hand, EAGLES (1996b) also includes the *recommended* units (i.e., *phrases*, *clauses* and *sentences*) and features associated to the (H)PSG formalism³³. This is a most remarkable weakness these recommendations, since they fail to address the (syntactic) annotations associated to other syntactic grammars, theories and/or formalisms, such as the functional or the dependency-oriented ones. Fortunately, the ISO Syntactic Annotation Framework standard has not neglected these other syntactic annotations and supplements perfectly EAGLES (1996b) recommendations.

The **ISO Syntactic Annotation Framework (ISO/SynAF) standard** (ISO/SynAF, 2010) tackles ‘the annotation of the syntactic constituency of such (groups of) morphosyntactically annotated fragments and the syntactic relations existing between those (groups of) morphosyntactically annotated fragments’ (ISO/SynAF, 2010). ISO/SynAF considers that the *sentence* defines the boundaries of the fragments of textual documents to which it is applied. As stated also in ISO/SynAF (2010), it ‘does not propose a tagset for syntactic annotation, but is dedicated to proposing a (possibly hierarchical) list of data categories, which is much easier to update and extend, and which will represent a point of reference for particular tagsets used for the syntactic annotation of various languages, also in the context of various application scenarios’ and is concerned ‘with a meta-model that covers both dimensions of syntactic constituency and dependency’. Therefore, the main guidelines for the design of a particular syntactic tagsets are still the EAGLES (1996b) recommendations.

3.2.1.3 SEMANTIC ANNOTATIONS

Semantic annotation is a multifaceted field. Meaning is difficult to be described and, as shown by the historical examples included at the beginning of this chapter (the determinatives of Egyptian writings and the Glosas Emilianenses), it has been annotated in many different ways, depending on the

³³ Most surprisingly, these recommendations regard no kind of syntactic annotation as *obligatory*).

(research) field. Consequently, many terms have appeared to refer to the same activity: *semantic annotation*, *semantic markup*, *semantic tagging* or *semantic labelling* are some of the terms found and seem to be used interchangeably. This subsection presents those semantic annotation-related terms, techniques and technologies coming from the linguistic field. The ones related to the Semantic Web are presented in Subsection 3.2.2.2.

3.2.1.3.1 Semantic Annotation Layers

Thus, to begin with, when speaking of *semantic annotation* or *semantic tagging*, the researchers in the linguistic field understand ‘semantic’ in various ways and, consequently, they annotate different aspects. McEnery & Wilson (2001) distinguish two broad types of semantic annotation in this field: (i) the semantic relationships holding between the items in the text (*i.e.*, the agents or patients of particular actions) and (ii) the annotation of word senses.

Regarding issue (i), it is referred to as **semantic role labelling** here. Basically, it consists in identifying and making explicit (*i.e.*, annotating) the predicate-argument structure of clauses and sentences. This predicate-argument structure links (or interrelates) the semantic projections of their syntactic constituents, that is, of their phrases and clauses, respectively. Each of these links is called a *semantic role*. A semantic role, for example, details the *agent* or the *patient* of a given *predicate* (an *action*) in a particular context. Thus, semantic role labelling clearly reveals the syntax-semantics interface of texts (Johnson & Fillmore, 2000; Gildea & Jurafsky, 2002; 2003; Kingsbury *et al.*, 2002).

As for the annotation of word senses (issue (ii)), several different types of semantic annotations have been proposed and implemented so far. Firstly, for Computational Linguistics, *semantic annotation* actually means **sense tagging**. This term refers to the association of content words in a text with their corresponding sense in an electronic dictionary or in other computational linguistic resource, *e.g.*, WordNet, or EuroWordNet for European languages (Kokkinakis & Kokkinakis, 1999). This approach is followed especially in what is known as *Word Sense Disambiguation* (WSD). This is a difficult annotation task, even if it is carried out manually. This difficulty derives mainly from the fact that the correspondence between words and senses (meanings) is not generally of the type one-to-one. Polysemy, homonymy, synonymy, contextual meaning and the impossible task of interpreting ambiguous words, if they are really ambiguous, are some of the problems commonly found. This has been extensively shown by the **Senseval initiatives**³⁴, carried out from 1998 to date, which include a series of tasks that aim(ed) at evaluating the strengths and weaknesses of (primarily) WSD programs with respect to different words, different varieties of language, and different languages (Kilgarriff, 1998; Kilgarriff & Rosenzweig, 2000).

³⁴ <http://www.senseval.org/>.

Secondly, there is a variant of WSD, namely *Word Domain Disambiguation (WDD)*, whose aim is to tag words in a text with a domain label in place of a sense label. This will be referred to as **domain annotation** here. The main advantage of domain annotation with respect to sense tagging is that domain labels reduce polysemy and, therefore, (automatic) domain annotation is usually more accurate than (automatic) sense tagging. In addition, several researches in the area argue that the results of applications like Information Retrieval (IR) and Question Answering (QA) are better when domain disambiguation is used instead of sense disambiguation (Suárez & Palomar, 2002; Sanfilippo *et al.*, 2006).

Thirdly, there is another very important application in the field of Information Extraction (IE) that is not referred to as either semantic or annotation, but which clearly provides text with meaning. More concretely, it can be thought of as a form of sense tagging for a particular type of terms appearing in texts that cannot be normally found in a common dictionary. These terms (mainly proper names) are referred to as *named entities*. **Named entities** can be defined as words or sequences of words that are ‘unique identifiers’ of *entities (organizations, persons, locations), time (dates, time), and quantities (monetary values, percentages)*³⁵. They are annotated by means of **Named Entity Recognition and Classification (NERC)** processes. Thus, the aim of a NERC process is to identify and categorise the set of named entities in a portion of text. The information they provide with their annotations is essential for the semantic interpretation of texts (Aguado de Cea *et al.*, 2009). Besides, as stated in the ISO/SemAF-NE (2009) new project proposal, the main areas of applications of named entity annotation are (a) information retrieval, (b) (semi-)automatic construction of ontologies, (c) automatic translation (d) message identification for automatic filtering, classification, and dispatching. As this document also stated, ‘all these applications share a common aim that is to improve information management by processing the content of the documents, notably in the context of the Semantic Web’.

Lastly, in Calzolari *et al.* (2001), semantic tagging stands for ‘the assignment, to corpus occurrences, of the appropriate semantic type/concept (such as *human, animal, etc.*) as defined within the SIMPLE project’ (described below). This type of tagging will be referred to as **semantic field annotation** (or tagging) henceforth, reusing a term included in Wilson & Thomas (1997), but with a slightly different sense.

Obviously, independently of the tagset chosen for each type of annotation, the following precedence relations hold:

$$\text{Tagset}_{\text{semantic field annotation}} \leq_{\text{granularity}} \text{Tagset}_{\text{domain annotation}} \leq_{\text{granularity}} \text{Tagset}_{\text{sense tagging}}$$

³⁵ http://www.cs.nyu.edu/cs/faculty/grishman/NEtask20.book_2.html#HEADING1.

Thus, domain-annotating a text implies its semantic field annotation, and sense-tagging a text implies both its domain annotation and its semantic field annotation. Accordingly, only one of these types of annotation is performed in each case, depending on the granularity of the task for which it is to be applied.

Hence, to summarise, the following layers of semantic annotation can be distinguished:

1. The **sense tagging layer**, whose goal is to assign an appropriate sense tag to each lexical unit included in a text, which describes precisely its meaning. It consists of two sub-layers, namely
 - a. The **concept semantic annotation layer**. The aim of this (sub)layer is to assign the appropriate sense to the content words in a text, using some kind of lexical resource. In Natural Language Processing (NLP), this task is known as *Word Sense Disambiguation (WSD)*. It is classified as an intermediate task, essential for Information Retrieval (IR) or Machine Translation (MT).
 - b. The **instance semantic annotation layer** (or **named entity annotation layer**). The aim of this (sub)layer is to identify, categorise and label the set of individuals in a portion of text. Names convey a lot of semantics. Together with nouns and pronouns, they are the way to refer to entities in text. In NLP, this task is known as *Named Entity Recognition and Classification (NERC)*.
2. The **semantic domain annotation layer**, in charge of tagging each semantic unit in a text with the domain to which it belongs.
3. The **semantic field annotation layer**, which aims at attaching a label to every word in a text to indicate the semantic field in which it falls.
4. The **semantic role labelling layer**, responsible for the annotation of the relationships established, in the syntax-semantics interface, between predicates and entities (as the annotations provided in the FrameNet project, briefly described below).

After the different types of semantic annotation that exist in both Corpus Linguistics and Computational Linguistics have been surveyed, it is the time to present the main projects, guidelines and standardisation initiatives related to this level of annotation. This is done in the following subsection.

3.2.1.3.2 Semantic Annotation-Related Projects, Guidelines and Standardisation Initiatives

No overall semantic annotation guidelines or standard have been published yet, *although in ISO/SemAF-Time (2009) it is stated that it is envisaged to develop a standard ‘concerned with semantic annotation and representation from an integrative point of view, dealing with the combined*

annotation of semantic information from several areas'. Thus, the different projects, guidelines and standardisation initiatives related to (linguistic) semantic annotation are partial and focus in a particular view of semantics. They are presented in this section according to the semantic annotation layer to which they can be related.

3.2.1.3.2.1 REGARDING THE SENSE TAGGING LAYER

The following projects and initiatives relating sense tagging must be mentioned: the WordNet and the EuroWordNet projects and the ISO Semantic Annotation Framework – Time and Events standard proposal.

As for the **WordNet and the EuroWordNet** projects, they gave birth to the most prominent and most commonly used lexical resources in the area of word sense tagging, namely WordNet and EuroWordNet.

On the one hand, **WordNet**³⁶ is a lexical database for the English language although, according to other authors, WordNet should be considered a linguistic ontology (Gómez-Pérez et al., 2004). WordNet was intended (i) to produce a combination of dictionary and thesaurus that is more intuitively usable, and (ii) to support automatic text analysis and artificial intelligence applications. It groups (American) English words into sets of synonyms, the so-called *synsets*, and records the various semantic relations between these synonym sets.

WordNet contained by 2006 about 150,000 words organized in over 115,000 synsets for a total of 207,000 word-sense pairs. For the constitution of its synsets, WordNet distinguishes between nouns, verbs, adjectives and adverbs, because they follow different grammatical rules. Every synset contains a group of synonymous words or collocations³⁷; thus, different senses of a word are in different synsets. The meaning of the synsets is further clarified with short defining *glosses* (*i.e.* definitions and/or example sentences).

Most synsets are connected to other synsets via a number of semantic relations. These relations vary, based on the type of word, and include the ones shown in table 3.

Although WordNet contains a sufficiently wide range of common words, one of its limitations is that it does not cover special domain vocabulary, since it is primarily designed to act as an underlying database (or ontology) for different (general-purpose) applications.

³⁶ <http://wordnet.princeton.edu/>.

³⁷ As for this purpose, a collocation in WordNet is a sequence of words that go together to form a specific meaning, such as “car pool”.

Table 3: WordNet relations holding between its different types of synsets

NOUNS	<i>hypernyms</i> : Y is a hypernym of X if every X is a (kind of) Y (<i>canine</i> is a hypernym of <i>dog</i>)
	<i>hyponyms</i> : Y is a hyponym of X if every Y is a (kind of) X (<i>dog</i> is a hyponym of <i>canine</i>)
	<i>coordinate terms</i> : Y is a coordinate term of X if X and Y share a hypernym (<i>wolf</i> is a coordinate term of <i>dog</i> , and <i>dog</i> is a coordinate term of <i>wolf</i>)
	<i>holonym</i> : Y is a holonym of X if X is a part of Y (<i>building</i> is a holonym of <i>window</i>)
	<i>meronym</i> : Y is a meronym of X if Y is a part of X (<i>window</i> is a meronym of <i>building</i>)
VERBS	<i>hypernym</i> : the verb Y is a hypernym of the verb X if the activity X is a (kind of) Y (<i>to perceive</i> is a hypernym of <i>to listen</i>)
	<i>troponym</i> : the verb Y is a troponym of the verb X if the activity Y is doing X in some manner (<i>to lisp</i> is a troponym of <i>to talk</i>)
	<i>entailment</i> : the verb Y is entailed by X if by doing X you must be doing Y (<i>to sleep</i> is entailed by <i>to snore</i>)
	<i>coordinate terms</i> : those verbs sharing a common hypernym (<i>to lisp</i> and <i>to yell</i>)
ADJECTIVES	<i>related nouns</i>
	<i>similar to</i>
	<i>participle of verb</i>
ADVERBS	<i>root adjectives</i>

On the other hand, **EuroWordNet**³⁸ is both a generalisation and the adaptation of WordNet to several different European languages, *i.e.*, Dutch, Italian, Spanish, German, French, Czech and Estonian. Hence, it is a multilingual database (or ontology) that consists of a number of interrelated wordnets (one for each of the languages that it includes). The wordnets are structured in the same way as WordNet, that is, in terms of synsets (sets of synonymous words) and the basic semantic relations that hold among them.

Each wordnet represents a unique language-internal system of lexicalisation. In addition, the wordnets are linked to an Inter-Lingual Index (ILI), based on WordNet. Through this Inter-Lingual Index, the languages are interconnected so that it is possible to go from the words in one language to similar words in any other language. The ILI also gives access to a shared top-ontology of 63 semantic categories (the SIMPLE Top Ontology, presented below). This top-ontology provides a common semantic framework for all languages, while language specific properties are maintained in the individual wordnets.

The cooperative framework of EuroWordNet has continued through the Global WordNet Association. This is a free and public association that builds on WordNet and EuroWordNet. Its aims are (i) to stimulate further building, standardization and interlinking of wordnets, (ii) the development

³⁸ <http://www.illc.uva.nl/EuroWordNet/>; <http://www.ilc.cnr.it/viewpage.php/sez=ricerca/id=820/vers=ing>.

of wordnet-based and wordnet-related tools and (iii) the dissemination of information. Thus, many institutes and research groups are developing similar wordnets for other languages (European and non-European) using the EuroWordNet specification (that is, the design of the database, the defined relations, the top-ontology and the Inter-Lingual Index). If compatible, these wordnets can be added to EuroWordNet and, through the index, connected to any other compatible wordnet.

There again, as for the **ISO Semantic Annotation Framework – Time and Events (ISO/SemAF-Time) standard proposal** (ISO/SemAF-Time, 2009), it provides normative guidelines for the annotation of temporal information, as well as for the annotation of various types of events (mainly in English, but also in other languages). Besides, this standard proposal defines a formal language for annotating temporal and event expressions, the so-called **ISO-TimeML**. Thus, whereas ISO/SemAF-Time as a whole provides a standard, consisting of basic concepts and a metamodel and others, ISO-TimeML is the recommended annotation language specified for ISO/SemAF-Time-conformant annotations.

Thus, ISO-TimeML has been devised to annotate all temporal objects, broadly categorized as temporal expressions and events. These temporal objects participate in temporal relationships (*e.g.*, *before* or *simultaneous*), subordinating relationships (*e.g.*, *intensional* or *factive*), and aspectual relationships (*e.g.*, *initiates* or *continues*). Specifically, four basic problems in event-temporal identification have been addressed in the design of ISO-TimeML and, hence, also in ISO/SemAF-Time: (1) time anchoring of events (identifying an event and anchoring it in time); (2) ordering events with respect to one another (distinguishing lexical from discourse properties of temporal ordering); (3) reasoning with contextually underspecified temporal expressions (temporal functions such as last week and two weeks before); (4) reasoning about the persistence of events (how long does an event or the outcome of an event last); and (5) handling basic tense and aspect features.

As can be observed, *the type of sense tagging standardised within ISO/SemAF-Time requires being complemented with the addition of the types of annotations provided by WordNet and EuroWordNet for sense-tagging the semantic units or discourse entities that participate in temporal or event expressions.*

Secondly, **concerning the instance semantic annotation (sub)layer** of the sense tagging layer, the following projects and initiatives must be mentioned: the Message Understanding Conferences series, the ACE program and the ISO/TC 37/SC4 Semantic Annotation Framework – Named Entities project.

As for the **Message Understanding Conference (MUC)** series, they were sponsored by the US Government to provide an assessment of the state of the art in the automatic processing of (short) messages, such as e-mails. The first work in the NERC area was performed in 1995 (Stevenson &

Gaizauskas, 2000), in the context of MUC-6, the sixth edition of the Message Understanding Conferences. Table 4 shows an excerpt of a MUC-6 named entity annotated text in SGML.

As with the Senseval initiatives, for each edition of the MUC, participants were given sample messages and instructions on the type of information to be extracted and, consequently, they developed (automatic) systems to process these messages. The participants had to submit afterwards the evaluations of their systems, based on these sample messages.

Table 4: An SGML-encoded named entity annotation example.

<pre>"It's a chance to think first level questions", said Ms. <enamel type="PERSON"> Cohn </enamel>, a partner in the <enamel type="ORGANIZATION"> McGlashan & Sarrail </enamel> firm in <enamel type="LOCATION"> San Mateo </enamel>, <enamel type="LOCATION"> California </enamel></pre>
--

The MUC series promoted traditional IE systems that were applied to limited and well defined domains. In fact, most of the work was domain-specific and language-dependent, since they focused on English journalistic corpora.

In MUC-6 and MUC-7 (Chinchor, 1997), seven different classes of NEs were identified, according to three main groups: person, organization and location. This classification, shown in Table 5, constitutes the main result of this conference series as for the semantic annotation layer described here. It has become a *de facto* standard in the field, thus being included in the ISO standard for named entity annotation described below.

As for the **ACE (Automatic Content Extraction) program**, it started in 1999 and promotes a deeper level of semantic processing than the one considered in MUC (*cf.* Maynard *et al.*, 2003). The ACE research aims at the detection and characterization of entities, relations, and events, not simply names. It also widens the domains covered by MUC and develops finer-grained classifications (Kokkinakis, 2004). *The main contributions of ACE* (Maynard *et al.*, 2003) are: (1) *it subdivided Location Named Entities into three different subclasses*, also shown in Table 5; and (ii) *it leads directly into the next level of linguistic annotation, discourse annotation, since it is linked to co-reference resolution*. With the task known as Entity Detection and Tracking (EDT), all mentions of an entity, whether a name, a description, or a pronoun, are to be found and collected into equivalence classes based on reference to the same entity (Doddington *et al.*, 2004).

As for the **Semantic Annotation Framework – Named Entities (ISO/SemAF-NE)** project proposal (ISO/SemAF-NE, 2009), it aims at proposing a consensual annotation scheme for Named Entities (NEs). The current specification of this standard deals with two aspects of an NE: its intrinsic identification and the extrinsic relations held by this NE.

Table 5: MUC-7 and ACE classifications (tagsets) of Named Entities.

MUC-7 CLASSIFICATION (TAGSET)	ACE CLASSIFICATION (TAGSET)	
ENAMEX: Entity (as such)	Person	Person
	Organisation	Organisation
	Location	Location
		Geo-Political Location (essentially, any kind of Location that has a government, such as a city or a country)
		Facility
NUMEX: Numerical expression (Quantities)	Monetary value	
	Percentage	
TIMEX: Temporal expression (Time)	Date	
	Time	

As stated also in ISO/SemAF-NE (2009), on the one hand, **the intrinsic identification of an NE** comprises the different elements that describe the NE, including: (a) its *semantic type*, such as *individual* or *organization*; (b) the *source type* that describes how the NE was recognized, such as *lexicon* or *pattern-based*; (c) the *kind of word form* used in the occurrence of the NE in the text, such as *abbreviation* or *full form*; and (d) the *structure* that decomposes the NE into substructures like, for instance, *given name* and *family name*. On the other hand, the **extrinsic relations held by the NE** describe the components in terms of a stand-off annotation. In other words, it is the set of links that allow referring to the annotation of the NES at other annotation levels, like the morphosyntactic and syntactic annotations of their constituent words or syntactic groups.

However, this standard specification fails to address ‘the mechanism that deals with co-reference [...] Notably, the annotation of the variants [of an NE] is not addressed where the challenge is to link the named entity occurrence "Jacques Chirac" with another occurrence like "J. Chirac". The co-reference from a pronoun to a named entity is not addressed.’ (*id.*).

3.2.1.3.2.2 REGARDING THE SEMANTIC DOMAIN ANNOTATION LAYER

Only the **SIMPLE** project has already dealt with this type of semantic annotation towards its standardisation. The **SIMPLE** project (SIMPLE, 2000), is a *follow up to PAROLE* that aimed at adding a semantic layer to a subset of the morphological and syntactic layers resulting from PAROLE. **PAROLE**, in turn, was the first project producing corpora and lexica in so many languages³⁹ and built according to the same design principles, linguistic specifications and representation format (Ruimy *et al.*, 1998). The PAROLE monolingual lexica and monolingual corpora consisted, respectively, of (i) 20,000 entries providing morphological and syntactic information and (ii) at least 20 million words for

³⁹ Catalan, Danish, Dutch, English, Finnish, French, German, Greek, Italian, Portuguese, Spanish and Swedish.

14 languages⁴⁰. Information was encoded following essentially the CES (Corpus Encoding Standard) designed by EAGLES⁴¹, on the basis of the TEI guidelines. 250,000 running words were tagged and checked at the morphosyntactic level, according to language specific instantiations of the EAGLES guidelines. The compatibility of the various corpora was ensured by the adoption of commonly defined criteria for composition, encoding and linguistic annotation.

The semantic lexicons (covering about 10,000 word meanings, 7,000 for nouns, 2,000 for verbs and 1,000 for adjectives) were being built in a *harmonised* (i.e., *standardised*) way for all the 12 languages covered by PAROLE. The main types of information encoded for nouns, verbs, and adjectives were: domain information, the semantic type of the head (with a structured semantic type), and the semantic type of the arguments of predicates (defined at different levels of granularity).

There again, *the SIMPLE project represented the first attempt to tackle the (standardised) encoding of semantic (argument) frames on a large scale, i.e.* for so many languages and with rather wide coverage. It also provided a framework for testing and evaluating the maturity of the by then current state-of-the-art in the realm of lexical semantics grounded on, and connected to, a syntactic foundation.

The main result of SIMPLE as for semantic domain annotation is the extensive hierarchy of domains that was identified and encoded within this project. It is included in SIMPLE (2000) and it was included also in the development of the present work (see Table 56 in page 126 – it is not reproduced here as well for the sake of space).

In addition, **a set of supplementary criteria for choosing or devising a semantic domain annotation⁴² tagset was proposed by Schmidt** (1988) and mentioned in Wilson & Thomas (1997). These criteria are the following:

1. *It should make sense in linguistic terms.* In other words, it must be formulated using the basic categories that exist in the mind, revealed by several psycholinguistic experiments thus far, such as colours, body parts or topography.
2. *It should be able to account exhaustively for the vocabulary in the corpus, not just for a part of it.*
3. *It should be sufficiently flexible to allow for those emendations that are necessary for treating a different period, language, register or textbase.*
4. *It should operate at an appropriate level of granularity (or delicacy of detail).*
5. *It should, where appropriate, possess a hierarchical structure.*

⁴⁰ Catalan, Belgian-French, Danish, Dutch, English, Finnish, French, German, Greek, Irish, Italian, Norwegian, Portuguese and Swedish.

⁴¹ <http://www.cs.vassar.edu/CES/>. For an XML implementation of this standard, see also <http://www.xces.org/>.

⁴² Referred to as a semantic field annotation in the sources cited.

6. *It should conform to a standard, if one exists.*

3.2.1.3.2.3 REGARDING THE SEMANTIC FIELD ANNOTATION LAYER

Only the EuroWordNet and the SIMPLE projects (presented in Section 3.2.1.3.2.1 and Section 3.2.1.3.2.2, respectively) have already dealt with this type of semantic annotation towards its standardisation.

On the one hand, the **EuroWordNet** project, as mentioned in Section 3.2.1.3.2.1, included an ontology⁴³, namely the **EuroWordNet Top-Ontology**. The first release of this Top-Ontology (Vossen *et al.*, 1998) consisted of 63 higher-level concepts. Following Lyons (1977), it distinguished 3 types of entities at the first level, namely (a) *first order entities*, that is, *concrete entities* (publicly) perceivable by the senses and located at any point in time, in a three-dimensional space; (b) *second order entities*, *i.e.*, those *static situations* (such as a *property* or a *relation*) or *dynamic situations* (b.1) that cannot be grasped, heard, seen or felt as an independent physical thing, and (b.2) that can be located in time and occur or take place rather than exist (*e.g.*, continue, occur or apply); and (c) *third order entities*, that is, unobservable propositions (c.1) that exist independently of time and space, (c.2) that can be true or false rather than real, and (c.3) that can be asserted or denied, remembered or forgotten (*e.g.*, idea, though, information, theory or plan).

The **SIMPLE** project reused the results the EuroWordNet Top-Ontology, conveniently ‘cleaned’, re-structured and adapted to its purposes (*cf.* SIMPLE, 2000). The result of this process (the main contribution in this respect of the project) is the top ontology formalised the SIMPLE semantic type system, which was subdivided into three layers:

- **The Core Ontology**, consisting of those types which were identified as the central and common ones for the construction of the different lexicons. The types in the Core Ontology represent the highest nodes in the hierarchy.
- **The Recommended Ontology**. This part of the ontology consists of more specific types (lower nodes in the hierarchy), which provide a more granular organization of the word-senses.
- **The Language Specific Types**, that is, those more detailed types that may be created in order to organize a lexicon for language-, domain- or application-specific needs. These types were not provided as such in SIMPLE, and can be eventually added if their elaboration is consistent with the organization of the rest of the SIMPLE model.

⁴³ This concept is conveniently defined in Section 3.2.2.2.

Besides, this top ontology (whose concepts are shown in Table 6) was structured as the union of two different hierarchies, one for nouns and verbs (the ontology of events), and one for adjectives (which can be roughly described as an ontology of abstract properties). Whereas the ontology for adjectives is a rather simple one, since it only distinguishes two main super-classes (*intensional* and *extensional*) with its corresponding (direct) sub-classes, the ontology of events is more complex and specialised. Briefly, the ontology of events incorporates results from WordNet, EuroWordNet and Levin’s Classes. Its aim was to find a number of event classes that is richer than that of WordNet (with a total of 15 classes) and less detailed than Levin (234 classes in total).

Table 6: The semantic categories included in the SIMPLE Top-Ontology

3D Location	Container	Language	Physical Property (Entity)
Abstract Entity	Convention	Living Entity	Plant
Abstract Property (Entity)	Domain	Location	Profession
Air Animal	Drink	Material	Psychological Property (Entity)
Animal	Earth Animal	Measurement Unit	Quality
Area	Entity	Micro-organism	Representation
Artifact	Flavouring	Money	Role
Artifact Food	Flower	Moral Standards	Semiotic Artifact
Artifactual Area	Food	Movement-Of-Thought	Shape
Artifactual Drink	Fruit	Natural Substance	Sign
Artifactual Location	Furniture	Number	Social Property (Entity)
Artifactual Material	Geopolitical Location	Opening	Social Status
Artwork	Human	Organical Object	Substance
Building	Ideo	Other Artifact	Temporary Activity Agent
Clothing	Information	People	Time
Cognitive Fact	Institution	Persistent Activity Agent	Vegetal Entity
Color	Instrument	Physical Object	Vehicle
Concrete Entity	Kinship	Physical Power	Water Animal

The main consequence of this organisation is that, in SIMPLE, the same types are used for the encoding of verbs and nouns denoting events. In other words, the type system does not depend on the syntactic category with which a semantic unit is linguistically realised. Under this perspective, both the verb ‘arrive’ and the noun ‘arrival’ express a semantic unit belonging to the same semantic type, *i.e.* an event of directed motion.

As commented above, the concepts (*i.e.*, the semantic categories) included in the resulting two top-ontologies of the SIMPLE (2000) project are shown in Table 6. These semantic categories can be regarded as the recommended tagset that should be applied to the semantic field annotation of texts (and corpora).

3.2.1.3.2.4 REGARDING THE SEMANTIC ROLE LABELLING LAYER

Even though the semantic annotations added to the Penn TreeBank dealt with the predicate-argument structure annotation for verbs, participial modifiers and nominalizations (Kingsbury *et al.*,

2002), the most advanced project in this area is FrameNet (Narayanan *et al.*, 2003). Thus, FrameNet should be considered a (de facto) standard for this type of semantic annotation for the time being.

The goal of **FrameNet** is to obtain a body of semantically and syntactically annotated sentences which provide reliable information on the valences or combinatorial possibilities of each item in the frame and its philosophy relies on the key concept of ‘semantic frame’ (Lowe *et al.*, 1997). A semantic frame is a script-like structure of inferences, which are linked to the meanings of linguistic units (lexical units). Each frame identifies a set of frame elements (FEs), which are frame-specific semantic roles (participants, properties, phases of a state of affairs). The frames can be simple –*i.e.* small static scenes or states of affairs, relations between entities and the roles they serve– or quite complex event types that provide the background for words.

So far, the results of the FrameNet project are included in what the authors call the FrameNet database, which incorporates the linguistic knowledge associated with lexical units. The description of each lexical unit identifies the frames which underlie a given meaning and the ways in which the FEs are realised in structures headed by the word. Suitable examples of lexical units and FEs can be found in the FrameNet homepage⁴⁴.

The FrameNet database (derived from corpora) can function as a monolingual dictionary (Ruppenhofer *et al.*, 2002) and as a thesaurus. As a dictionary, each lexical unit is provided with the name of the frame it belongs to and access to a description of the frame. The database incorporates also a definition and a valence description, summarizing both the semantic roles and the syntactic form and function of the phrases that instantiate those roles. Access to annotated examples is also provided. As a thesaurus, words are linked to the semantic frames in which they participate and, hence, to the other words which evoke those frames.

At present, FrameNet is also being elaborated for Spanish (Subirats, 2004) within the **Spanish FrameNet (SFN)** project. The ‘starter lexicon’ of SFN is publicly available, and contains more than 1,000 lexical units (verbs, predicative nouns, and adjectives, adverbs, prepositions and entities) representative of a wide range of semantic domains (SFN, 2012).

3.2.1.3.3 Semantic Annotations – Concluding Remarks

In this section, we have surveyed semantic annotation with the aim of (1) clearing up some of the confusion found in the terminology used, (2) describing the main elements that constitute this (linguistic) annotation level, (3) presenting its related standards (including standard proposals and *de*

⁴⁴<https://framenet.icsi.berkeley.edu/fndrupal/>.

facto standards), and (4) identifying possible ways to integrate these types of (linguistic) semantic annotation with the annotations for the Semantic Web presented below. This last issue will be discussed in detail at the end of the present chapter. The first three have been summarised in Table 7 (included in page 55).

3.2.1.4 LINGUISTIC ANNOTATION: LEVEL-INDEPENDENT APPROACHES

Whereas the previous subsections survey the different levels, layers and approaches to linguistic annotation concerned by the present work, this subsection presents linguistic annotation from a holistic point of view. In other words, this section summarises the different recommendations, criteria, guidelines and standards dealing with linguistic annotation as a whole, from a global perspective and, in particular, with (a) the (formal) representation of linguistic annotations and (b) the elaboration of any kind of (linguistic) annotation scheme.

The first set of **level-independent guidelines for annotation** was **suggested by Leech** (1997) and also referenced in McEnery & Wilson (2001):

1. The original text should be easily *recoverable* by taking away the annotations added to it.
2. Annotations should be easily *extricable* from the annotated text.
3. Every annotated text must be accompanied with a thorough *documentation*, including, among others, the **annotation scheme** –the particular and precise guidelines used to annotate a text–, how (manually and/or automatically), by whom the text was annotated and the quality of the annotation (*e.g.*, an accuracy rate).
4. The corpus annotation is *not infallible*: any act of annotation is also an act of interpretation.
5. Annotation schemes should be based as far as possible on *consensual*, widely agreed and theory-neutral principles.
6. *No* annotation scheme should claim authority as an absolute *standard*.

In practice, these recommendations suggest some comments: currently, recommendations (1), (2) and (3) –to some extent– are easily fulfilled with the use of HTML, XML or similar mark-up languages⁴⁵, which allow for a **standoff annotation**, whereas the inclusion of recommendation (3) in an annotation scheme only requires indicating in its annotation guidelines (or manual) how these interpretations should be systematically and coherently made.

⁴⁵ These recommendations were made before the family of mark-up languages such as SGML, HTML and XML was fully developed.

Table 7: A summary of the different elements concerning (linguistic) semantic annotation

LAYER / SUBLAYER	RELATED PROJECTS & INITIATIVES		RELATED ELEMENTS	STABLE TAGSETS
	ISO (STANDARDISATION)	OTHER		
Sense Tagging Layer	<ul style="list-style-type: none"> The Semantic Annotation Framework – Time and Events (ISO/SemAF-Time) project (not general) 	<ul style="list-style-type: none"> The Senseval initiatives The WordNet project The EuroWordNet project 	<ul style="list-style-type: none"> (Content) <i>Words</i> <i>Senses</i> 	<ul style="list-style-type: none"> WordNet synsets EuroWordNet synsets The ISO-TimeML tagset (not general)
	<ul style="list-style-type: none"> The Semantic Annotation Framework – Named Entities (ISO/SemAF-NE) project 	<ul style="list-style-type: none"> The MUC series The ACE program 	<ul style="list-style-type: none"> <i>Names</i> <i>Named entities</i> 	<ul style="list-style-type: none"> The MUC-7 tagset The ACE tagset
Semantic Domain Annotation Layer		<ul style="list-style-type: none"> The SIMPLE project Schmid's criteria for semantic domain annotation 	<ul style="list-style-type: none"> <i>Words</i> <i>Semantic types/concepts</i> (such as <i>human, animal, etc.</i>) 	<ul style="list-style-type: none"> The hierarchy of domains included in SIMPLE (2000)
Semantic Field Annotation Layer		<ul style="list-style-type: none"> The EuroWordNet project The SIMPLE project 	<ul style="list-style-type: none"> <i>Words</i> <i>Semantic fields</i> 	<ul style="list-style-type: none"> The SIMPLE-EuroWordNet Top-Ontology
Semantic Role Labelling Layer		<ul style="list-style-type: none"> The FrameNet project The Penn Treebank semantic annotation project 	<ul style="list-style-type: none"> <i>Predicate-argument structures</i> <i>Semantic roles</i> (such as <i>agent or patient</i>) 	<ul style="list-style-type: none"> The frame elements The lexical units The semantic roles (identified in FrameNet)

SEMANTIC ANNOTATION

Recommendation (3) can be accomplished through the use of widely known ontologies or by the definition of some kind of standard, but this would prevent recommendation (6) from being fulfilled. Considering that research funding authorities are highly encouraging the unification and standardisation of annotation schemes through the EU EAGLES-related initiatives, and, more recently, the FLaReNet⁴⁶ network, or the US SILT⁴⁷ project, it seems necessary that recommendation (6) be interpreted in a more relaxed way⁴⁸.

In addition to Leech's practical guidelines, for some of the annotation levels and layers introduced in the previous sections, the **EAGLES** project reached a consensus on three aspects: what to annotate, to what extent and how. This consensus was expressed by means of some **principles**, which underlie the EAGLES (1996a; 1996b) recommendations:

1. Linguistic annotation schemes should make use of an attribute-value formalism.
2. In addition, they should consider three constraint sublevels (*obligatory*, *recommended* and *optional*) in defining what is acceptable according to the guidelines.
 - **Obligatory annotations** are crucial for the phenomenon being annotated. Thus, they are required in the annotation scheme associated to that phenomenon for this reason and in order to be conformant with EAGLES guidelines.
 - **Recommended annotations** are not required, but should not be omitted. If these recommended attributes and values occur in a particular language, then it is advisable that the tagset of that particular language should encode them.
 - **Optional annotations** are neither required nor recommended, but specific to a (set of) language(s) or a language engineering application.

Finally, *ISO/TC 37* is also concerned with the elaboration of global standards for linguistic annotation. This concern is shown by *the ISO Linguistic Annotation Framework and the ISO Data Category Registry standard drafts* (i.e., they standards under development).

The **ISO Linguistic Annotation Framework (ISO/LAF)** 'is designed to support the development and use of computer applications for linguistic annotations and representations and the exchange of such data between different applications' (ISO/LAF, 2009). It also 'specifies a model that has been designed for the purpose of providing guidance on the basic principles for representing linguistic resources annotation schemes' (*id.*).

⁴⁶ Fostering Language Resources Network (<http://www.flarenet.eu/>).

⁴⁷ Sustainable Interoperability for Language Technology (<http://anc.cs.vassar.edu/SILT/>).

⁴⁸ As for the notion of standard, EAGLES (1996b) states: "there is no absolute normative prescription of annotation practices, but at most a set of recommendations (criteria) from which the annotator may justify departures or extensions for particular purposes".

ISO/LAF is built around some relatively straightforward ideas: (a) the separation of information conveyed by means of structure and information conveyed directly by specification of content categories; (b) the development of an abstract format that puts a layer of abstraction between site-specific annotation schemes (referred to as annotation schemas here) and standard specifications; and (c) the creation of a *Data Category Registry* (described below) to provide a reference set of annotation categories (related to (a) above). This **defines a clear separation between the linguistic content of annotation schemes and the way in which annotations are represented or encoded**.

Hence, with this aim, ISO/LAF specifies an abstract model for annotations instantiated by a pivot format, onto and out of which annotations are mapped for the purposes of exchange. To map to the pivot, an annotation scheme must be (or be rendered, via the mapping) isomorphic to the abstract model, which consists of (1) a referential structure (instantiated as a directed graph) for associating stand-off annotations with primary data; and (2) a feature structure representation for annotation content (*i.e.*, following the EAGLES principles previously mentioned). **In ISO/LAF, thus, an annotation forms a directed graph, in which nodes are labelled with feature structures providing the annotation content**⁴⁹.

However, ISO/LAF (2009) ‘does not provide specifications for annotation content categories (*i.e.*, the contents of the associated linguistic phenomena, or, in other words, about concrete linguistic annotation tagsets)’. Instead, the ‘standardization of data categories and methods for the specification of data structures are given in ISO 12620’ (that is, in the *Data Category Registry*).

Accordingly, the **Data Category Registry (ISO/DCR)**⁵⁰ aims at identifying, collecting and formally defining (that is, standardising) a set of linguistic categories in common use within the language engineering community (Wright, 2004; ISO/DCR, 2008). In other words, it **aims at standardising the different tagsets that should be used for the linguistic annotation levels and layers presented above**. Accordingly, this formally defined set of categories will ‘provide (1) a precise semantics for annotation categories that can be either used “off the shelf” by annotators or modified to serve specific needs; (2) a set of reference categories onto which scheme-specific names can be mapped; and (3) a point of departure for the definition of variant, more precise, or entirely new data categories for use in language resource annotation.’ (*id.*).

Therefore, ISO/LAF and ISO/DCR provide complementary views of linguistic annotation: while ISO/LAF deals with the specification of a format of representation and/or a(n) (abstract) scheme for all kinds of linguistic annotation that are both standardised and integrative, ISO/DCR aims at

⁴⁹ This graph model underlies also Semantic Web formats such as RDF and OWL (presented below), so that any graph can be trivially represented (or, in the jargon of the area, implemented or *serialised*) by means of these languages.

⁵⁰ <http://www.isocat.org/>.

providing concrete standardised tagsets for their instantiation, according to each particular linguistic annotation level and layer.

This concludes the survey of of linguistic annotation. It is summarised in Table 8 (included in page 59). The computational approach to annotation is presented in the following section.

3.2.2. THE COMPUTATIONAL APPROACH TO ANNOTATION

The main and most remarkable characteristic of a computational approach to annotation (especially when compared to the historical and the traditional linguistic approaches presented above), is that annotations are added having a very different target (*i.e.*, type of reader) and purpose in mind than the one expected to read ancient texts and their annotations. That is, in a computational approach, the targets of annotations are machines, instead of people. Therefore, in order for machines to understand (or, in other words, to process) these annotations, they must be encoded using most formal and unambiguous vocabularies, languages and/or schemas.

3.2.2.1 COMPUTATIONAL REPRESENTATION OF ANNOTATIONS: ANNOTATION LANGUAGES

The need for (standard) computational annotation languages was originated by the appearance of the World Wide Web (the WWW or, simply, the Web). The spreading of the WWW required encoding, publishing, handling (*e.g.*, presenting) its documents in a common and uniform (*i.e.*, standard) way.

Therefore, a whole family of annotation languages (together with their corresponding vocabularies and schemas) was devised for detailing both the content and the associated formal characteristics (encoded as annotations) of the documents added to and/or referenced in the WWW. These annotation languages were called initially markup languages. The ancestor of all these markup languages was IBM's **Generalised Markup Language (GML)**, which was designed in the 1960s to enable the sharing of machine-readable large-project documents in government, law, and industry⁵¹.

⁵¹ http://en.wikipedia.org/wiki/Standard_Generalized_Markup_Language.

Table 8: A summary of the different elements concerning linguistic annotation

LEVEL	RELATED PROJECTS & INITIATIVES		RELATED ELEMENTS	TAGSETS
	ISO	OTHER		
Linguistic Annotation	Level-Independent	ISO/LAF, ISO/DCR	Leech's criteria, EAGLES principles	<ul style="list-style-type: none"> • ISO/DCR
	Morphosyntactic Level	ISO/MAF	EAGLES (1996a)	<ul style="list-style-type: none"> • EAGLES (1996a) • ISO/DCR – morphosyntactic level
	Syntactic Level	ISO/SynAF	EAGLES (1996b)	<ul style="list-style-type: none"> • EAGLES (1996b) • ISO/DCR –syntactic level
Semantic Level	See Table 7			

Later on, the International Organization for Standardization (ISO) completed in 1986 a standardised version of GML, namely the **Sandard Generalised Markup Language** (SGML). **SGML** details a framework for the definition of generalized markup languages for documents. SGML promotes, amongst others, (i) the separation of logical and physical structures (elements and entities), (ii) the separation of data and metadata (elements and attributes), (iii) a mixed content, and (iv) the default angle-bracket syntax, that is, SGML elements consisting of markup tags surrounded by angle brackets⁵². Besides, the requirements mentioned in this standard for a markup to be considered a generalized markup are that (i) it should describe a document's structure and other attributes, rather than specify the processing to be performed on it and (ii) it should be rigorous.

In these properties and requirements mentioned above, (1) *entities* can be loosely defined as the content of the document that is marked up (*i.e.*, annotated); (2) the *markup tags* encode the annotations that are added to entities; (3) the *elements* are the data structures by means of which entities are added the annotations within the document; (4) *attributes* are the metadata that characterise or comment other properties of the entity in question, (5) a *mixed content* means that it is proposed to keep both the document original content and its annotations together, and (6) the *default angle-bracket syntax* is the language mechanism postulated to differentiate the tags (or the annotations) from the content of the document.

Most likely, the most well-known descendants of SGML, which fulfil the properties and the requirements mentioned above, might be the **HyperText Markup Language** (HTML) and the **Extensible Markup Language** (XML).

Concerning **HTML**, this markup language provides a means to create structured documents by denoting structural semantics for text, such as headings, paragraphs, lists, etc., as well as for links, quotes, and other items⁵³. In 2000, HTML also became an international standard. It is nowadays the predominant text and image formatting language used by Web browsers to dynamically format Web pages. As promoted by its ancestor, SGML, HTML documents are written in the form of HTML elements consisting of 'tags' surrounded by angle brackets, which structure the Web page (content).

The formulation and standardisation of both SGML and HTML meant an important step forward in the definition of annotations and annotation languages. On the one hand, for the first time ever, annotations could be added not only to texts, but also to images, program (Java) scripts, etc. This entailed that the concept of document content and the range of the elements of a document that could be annotated were clearly widened. **On the other hand, it enabled the**

⁵² <http://en.wikipedia.org/wiki/XML#Sources>.

⁵³ *cf.* <http://en.wikipedia.org/wiki/HTML>.

development of computational tools that could (i) produce annotations (semi)automatically and (ii) process these annotations automatically, at least for presentational purposes.

Nevertheless, as can be easily deduced, the intended use of HTML got a bit away from the intended use of the annotations included in the previous section. Whereas the latter HTML annotations were used to comment or explain somehow the text to which they were added, the former are intended to detail the presentational and layout properties of Web document (or pages) content. This can be considered a very practical but also too narrow a view to annotate Web pages. That is why other SGML-derived markup languages were designed. The aim of these other markup languages was to add other types of computational annotations to documents as, for example, annotations that described the content of documents itself and/or explained its meaning (as in the Semantic Web, for instance). Thus, XML and the family of XML-based languages were born.

Regarding **XML**, it is a profile (that is, a reworking) of SGML, and most of XML comes from SGML unchanged. It is defined in the XML 1.0 Specification⁵⁴ produced by the World Wide Web Consortium (W3C) and several other related specifications⁵⁵. XML emphasises simplicity, generality, and usability over the Internet. Although these design goals focus on documents, XML is widely used for the representation of arbitrary data structures, for example in Web services. In fact, **nowadays, XML is being used to annotate not only Web documents and their content, but also Web services and other Web resources**, so that they can be conveniently located and (re)used.

However, as stated above, XML is only a set of rules for encoding documents. Therefore, it should be regarded of as an annotation meta-language. Thus, XML can and must be completed or complemented (with a suitable vocabulary, for example) in order to be considered a proper annotation (or markup) language and, in particular, an annotation language suitable for the annotation of the semantics or the meaning of Web resources and content. This is the origin of the next markup language presented in this section (RDF) and of the two markup languages described in Subsection 3.2.2.2.1.

The **Resource Description Framework (RDF)** is a family of W3C specifications originally designed as a metadata data model⁵⁶. It has come to be used as a general method for the conceptual description (or modelling) of the information that is implemented in Web resources, using a variety of syntax formats.

⁵⁴ <http://www.w3.org/TR/REC-xml/>.

⁵⁵ <http://www.w3.org/>.

⁵⁶ http://en.wikipedia.org/wiki/Resource_Description_Framework.

The **RDF data model is based upon the idea of making statements about resources (in particular Web resources) in the form of <subject, predicate, object> triples.** In these triples, (i) the *subject* denotes the resource, (ii) the *predicate* denotes a property (or a characteristic) of the *subject* or expresses a relationship between the *subject* and the *object* and (iii) the *object* identifies the value of the property in question for the *subject* or the resource linked to the *subject* by means of the relationship identified by the predicate. For example, the statement ‘John bought an apple’ could be represented by means of an RDF *triple* in which ‘John’ would be the *subject*, ‘bought’ would be the *predicate* and ‘an apple’ would be the *object*.

As for annotation purposes, the *subject* of an RDF *triple* could be viewed as the content or the resource that is being annotated, the *predicate* would be the characteristic or the property that needs to be annotated and the *object* could be regarded as the tag added to comment or explain the value of this property for the *subject* of the *triple*.

These three components of RDF *triples* (the *subject*, the *predicate* and the *object*) are usually specified by means of **Uniform Resource Identifiers (URIs)**. A *URI* is a string of characters used to identify a name or a resource on the Web. Such identification enables the interaction between the different representations of the resource over a network (typically the WWW). A *URI* can be classified as a **Uniform Resource Name (URN)**, a **Uniform Resource Locator (URL)**, or both. Whereas a *URN* defines an item’s identity, a *URL* provides a method for finding it.

The main advantage of RDF is that a collection of RDF triples intrinsically represents a labelled, directed multi-graph. This allows for a direct application of well-known and already implemented directed multi-graph algorithms to the processing of RDF triple collections. In addition, (1) the design of linguistic annotation schemes following ISO/LAF (see Subsection 3.2.1.4) can be implemented almost straightforwardly using this language, since they share their underlying mathematical formalism (*i.e.*, directed (multi-)graphs) and (2) additional ontology languages (or vocabularies) can be built upon RDF, as RDFS and OWL demonstrate (see Subsection 3.2.2.2.1). In fact, the RDF mechanism for describing resources, together with the particular vocabularies defined by means of RDFS and/or OWL, are very important for the Semantic Web (the area of application of the hybrid annotation model discussed in this dissertation). This area is described in the following section.

3.2.2.2 THE SEMANTIC WEB AND SEMANTIC (WEB) ANNOTATIONS

Internet has become the main source of information nowadays. The amount of documents available in the World Wide Web (WWW) is enormous. In spite of search engines such as *Google*, *Yahoo*, or *AltaVista*, locating information on the WWW is very difficult for readers. In order to improve searches

and other web- and meaning-based processes, there is a need to create machine-readable content. This is what the Semantic Web (Berners-Lee & Fischetti, 1999) aims at: making texts understandable for computers. This need for documents in which content is machine-processable is what has made (semantic) annotation an essential activity for computer scientists.

According to Berners-Lee *et al.* (2001), up to then ‘most of the Web's content’ had been ‘designed for humans to read, not for computer programs to manipulate it meaningfully’. Computers could ‘parse Web pages for layout and routine processing’ but, in general, they had ‘no reliable way to process the semantics’. The Semantic Web was intended as an evolution of the Web that could ‘bring structure to the meaningful content of Web pages, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for users’.

Accordingly, following the same author, ‘the **Semantic Web** is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation’. Until then, the Web had ‘developed most rapidly as a medium of documents for people rather than for data and information that can be processed automatically. The Semantic Web aims to make up for this’. Figure 3 (included in the next page) summarises the main differences between the World Wide Web and its extension, the Semantic Web.

However, ‘for the Semantic Web to function, computers must have access to structured collections of information and sets of inference rules that they can use to conduct automated reasoning. [...] The challenge of the Semantic Web, therefore,’ was ‘to provide a language that expressed both data and rules for reasoning about the data and that allowed rules from any existing knowledge-representation system to be exported onto the Web’. This (annotation) language is responsible for attaching to the information in Web pages their well-defined meaning explicitly, transforming them into Semantic Web pages, which computers can read and process in a more intelligent way.

The development of such an annotation language involved the use of one of the basic components of the Semantic Web, namely ontologies. An **ontology** is a formal (and explicit) specification of a shared conceptualisation (Gruber, 1993; Borst, 1997; Studer *et al.*, 1998). Firstly, by formal, it is meant that it must be machine-readable. Secondly, being explicit implies that the type of concepts used and the constraints on their use have to be explicitly defined. Thirdly, as any other type of specification, an ontology consists of concepts, properties, relationships, functions, constraints, and axioms. Fourthly, it has to be shared, that is, it has to capture consensual knowledge. Finally, by virtue of being a conceptualisation, it constitutes an abstract model of some phenomenon in the world, which identifies the relevant concepts of this phenomenon.

The transition from the WWW to the Semantic Web

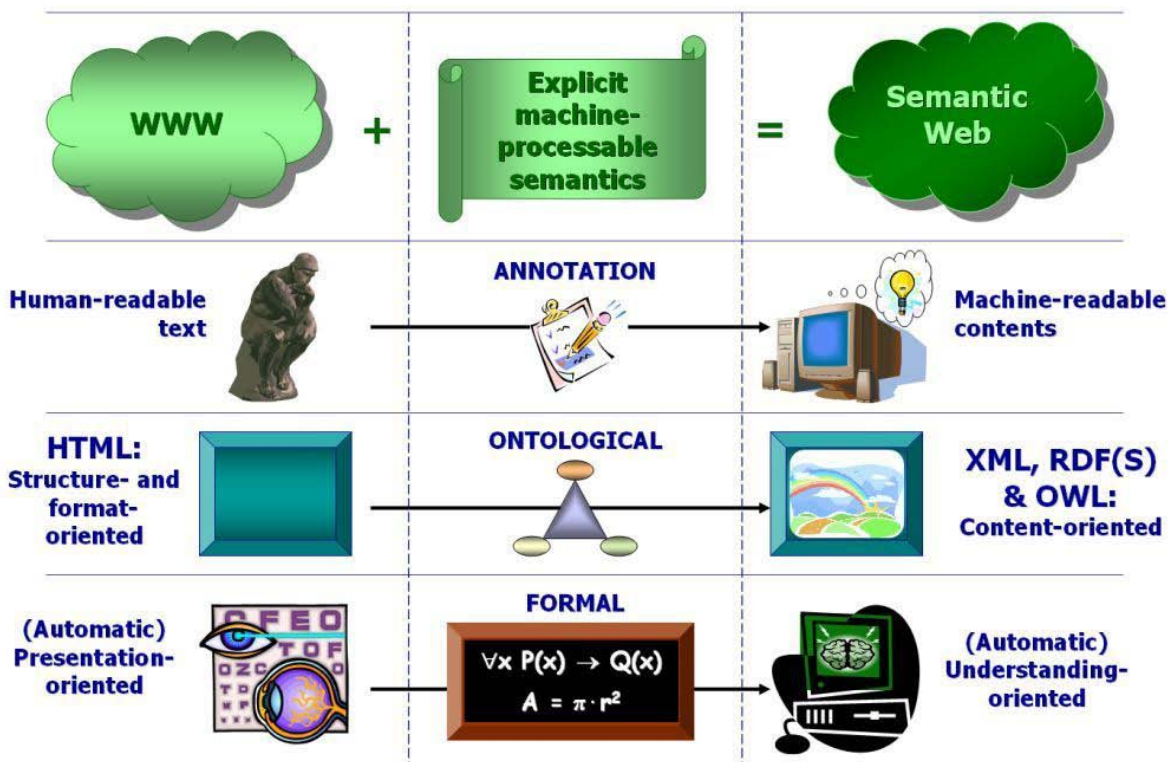


Figure 3: The transition from the WWW to the Semantic Web

There again, Berners-Lee *et al.* (2001) states that ‘the most typical kind of ontology for the Web has a taxonomy and a set of inference rules. The taxonomy defines classes of objects and relations among them’. ‘Classes, subclasses and relations’ are said to be ‘a very powerful tool for Web use’. With this tool we can express ‘a large number of relations among entities by assigning properties to classes and allowing subclasses to inherit such properties’.

As this author also states, ‘with ontology pages on the Web, solutions to terminology (and other) problems begin to emerge. The meaning of terms or XML codes used on a Web page can be defined by pointers from the page to an ontology’. From this point of view, **ontologies can be viewed as a way to define a formal vocabulary that expresses the meaning of terms in an explicit and machine-readable way**. In other words, ontologies and, more concretely, their components, can be viewed as the particular (formal and machine-readable) tagset associated to the annotations of the Semantic Web. They help make explicit the meaning of the information included in the Semantic Web documents (and/or pages). This is usually achieved by means of their (semantic) annotation. Hence, ontologies and ontology components (or terms) are to the Semantic Web what determinatives were to Egyptian texts and inscriptions. However, whereas determinatives were added for humans to read and process them, ontology-based semantic annotations are added for machines to read and process them.

Consequently, much research has already been carried out lately by ontological engineers on the semantic annotation of Web pages and other Web resources (Luke & Heflin, 2000; Benjamins *et al.*, 1999; Motta *et al.*, 1999; Staab *et al.*, 2000). As shown in Figure 4, the annotation of these Web resources with ontological information intends to allow intelligent access to Web pages, ease their searching and browsing and exploit new Web inference approaches from the information they contain.

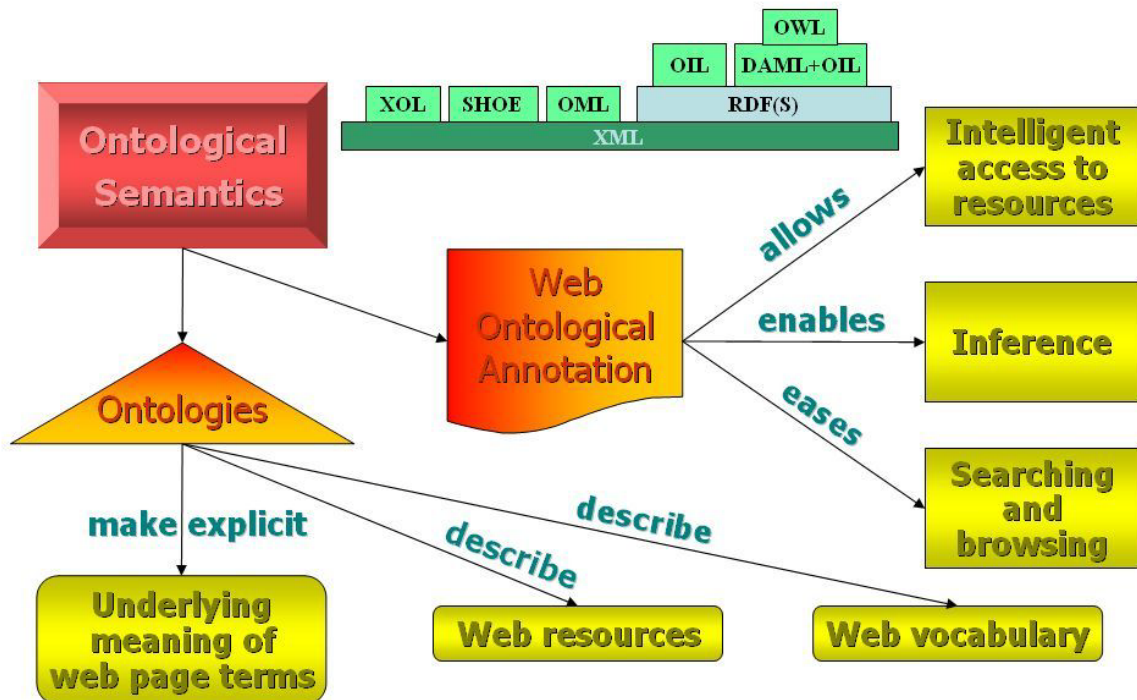


Figure 4: Ontologies and Semantic Web Annotations.

Many systems and projects on ontology-based annotation have been developed and carried out so far: SHOE (Luke *et al.*, 2000); the (KA)² initiative (Benjamins *et al.*, 1999); PlanetOnto (Motta *et al.*, 1999) and the Semantic Community Web Portals project (Staab *et al.*, 2000). Several semantic annotation tools have also been developed hitherto: COHSE (COHSE, 2002), MnM (Vargas-Vera *et al.*, 2001), OntoMat-Annotizer (OntoMat, 2002), SHOE Knowledge Annotator (SHOE, 2002) and AeroDAML (AeroDAML, 2002). For a comprehensive survey on ontology tools, see OntoWeb (2002). However, the main drawback of most of these tools is that they were conceived as (1) aiding tools for the human annotation of Semantic Web documents, instead of fully automatic annotation tools (unlike most linguistic annotation tools); or (2) machine-learning-based tools, with a poor accuracy or coverage (especially when compared with their related linguistic tools, *i.e.*, sense taggers or named entity recognition and subclassification systems).

Besides, a related theory has already been created, in order to support the development of this type of annotations from a linguistic point of view, namely Ontological Semantics. **Ontological Semantics** (Niremburg & Raskin, 2004) is a theory of meaning in natural language⁵⁷ which uses a constructed world model –the ontology– as the central resource for: (i) extracting and representing meaning of natural language texts; (ii) reasoning about knowledge derived from texts; and (iii) generating natural language texts based on representations of their meaning (see Figure 4).

A schematic example of use of ontologies for annotation in the Semantic Web has been included in Figure 5⁵⁸ (also on next page). It shows an attempt to integrate Named Entity Recognition and Classification with the annotations for the Semantic Web by means of the Knowledge and Information Management platform⁵⁹ (KIM). In this attempt (Kiryakov *et al.*, 2005) semantic annotation is ‘about assigning to the entities in the text links to their semantic description’.

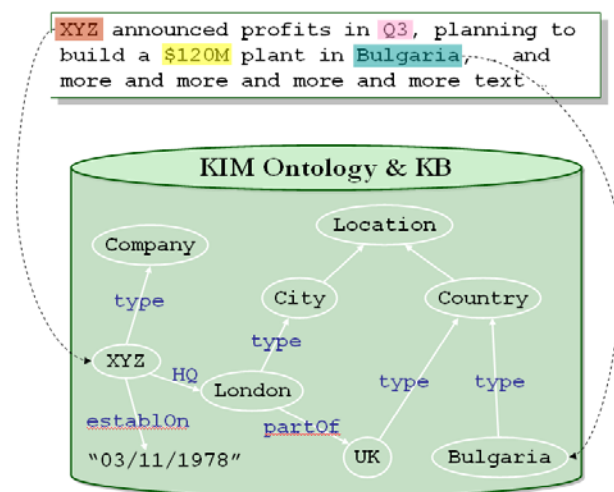


Figure 5: Semantic annotation in the Semantic Web – an example.

Nowadays, research in the Semantic Web field is oriented towards annotating⁶⁰ multimedia data (text, images, semi-structured information, voice, etc.) and multilingual data, in order to identify relevant documents written in several languages.

All these types of annotations are expressed by means of another basic component of the Semantic Web, that is, Semantic Web languages, which provide a way to both (i) represent annotations in a structured and formal way; (ii) develop formal vocabularies (or tagsets) for encoding these annotations. These Semantic Web languages are presented in the next subsection.

⁵⁷ And also an approach to natural language processing (NLP).

⁵⁸ Example taken from <http://www.ontotext.com/kim/semanticannotation.html>.

⁵⁹ <http://www.ontotext.com/kim/>

⁶⁰ For an extensive classification of the different types of annotations that can be applied to (Semantic) Web pages, see Bechhofer *et al.* (2002). A discussion on the degree of formalization for the different types of annotation can be found in Uschold (2003).

3.2.2.2.1 Definition of Ontological Tagsets and Metadata: RDFS and OWL

In the Semantic Web, (semantic) annotations have been carried out by means of mark-up languages such as XML and RDF (presented above), RDF Schema (Lassila & Swick, 1999; Brickley & Guha, 2000) or, lately, OWL (McGuinness & van Harmelen, 2003)⁶¹. **RDF Schema and OWL are two RDF extensions or particularisations used for encoding Semantic Web documents and resources.** Each of them can be considered a specific vocabulary that can be added to (or on top of) XML in order to constitute a full and proper (computational) annotation language.

Despite the powerfulness of RDF (described above), it does not fix at all the semantics associated to URIs, which are used for describing resources. Therefore, it is necessary for producers and consumers of RDF *triples* to agree on the semantics of resource identifiers. RDF Schema and OWL were created for this purpose, since the semantics of URIs of ontological terms can be made explicit by publishing on the Web their corresponding ontologies, implemented in any of these two RDF extensions. The final aim of this strategy is to establish, or circumscribe, the intended meanings of the URIs used to express data in RDF. Thus, for example, the URI <http://www.oeg-upm.net/OntoTag/LUO#SemanticUnit> could be used to refer to the class of all semantic units, considering that (a) *SemanticUnit* is a concept of an ontology (in this case, the *LUO*, that is, the Linguistic Unit Ontology – see Subsection 4.1.3); (b) this ontology has been implemented in OWL (though it could have been implemented in RDF Schema as well); and (3) it has been published at <http://www.oeg-upm.net/OntoTag>. **Besides, the RDF data model does not provide mechanisms for defining the relationships between properties (attributes) and resources. That is, precisely, the role of RDF Schema and OWL.** Let us see now these two languages in detail.

RDF Schema (RDFS, and also RDF(S)) is an extensible knowledge representation language, which provides basic elements for the description of ontologies, otherwise called RDF vocabularies, intended to structure RDF resources⁶². The first version of RDFS was published by the World Wide Web Consortium (W3C) in April 1998, and the final W3C recommendation was released in February 2004.

RDFS offers primitives for defining knowledge models that are closer to frame-based approaches. Thus, the **main RDFS constructs** are the following relations: (1) `rdfs:Class`, which declares a resource as a(n) (ontology) class for other resources; (2) `rdfs:subClassOf`, which allows to

⁶¹ For an extensive and detailed list of the markup languages used in the Semantic Web, see Gómez-Pérez & Corcho (2002).

⁶² http://en.wikipedia.org/wiki/RDF_Schema.

declare hierarchies of classes; (3) `rdf:Property`, which must be understood as the class of RDF properties (each member of this class is an RDF *predicate*); (4) `rdfs:domain` (associated to an `rdf:Property`), which declares the class of the *subject* in a *triple* whose second component is the *predicate*; (5) `rdfs:range` (associated to an `rdf:Property`), which declares the class or data type of the *object* in a *triple* whose second component is the *predicate*; and (6) `rdfs:subPropertyOf`, which is an instance of `rdf:Property` that is used to state that all resources related by one property are also related by another.

RDFS also declares the following utility properties: (1) `rdfs:seeAlso` is an instance of `rdf:Property` that is used to indicate a resource that might provide additional information about the subject resource; and (2) `rdfs:isDefinedBy` is an instance of `rdf:Property` that is used to indicate a resource defining the subject resource. This property may be used to indicate an RDF vocabulary in which a resource is described.

Finally, some other constructs are also predefined in RDFS, such as (1) `rdfs:label`, which is an instance of `rdf:Property` that may be used to provide a human-readable version of a resource's name; and (2) `rdfs:comment`, which is an instance of `rdf:Property` that may be used to provide a human-readable description of a resource, *i.e.*, it can be a statement in natural language.

Concerning the **Web Ontology Language (OWL)**, it is a family of (semantic) markup languages for publishing and sharing ontologies on the World Wide Web⁶³. This family of languages has also been endorsed by the World Wide Web Consortium and developed as a vocabulary extension of RDF⁶⁴. Thus, it incorporates many RDFS components. However, each of the languages of this family is said to be more expressive than RDFS. OWL is considered one of the fundamental technologies underpinning the Semantic Web, and has attracted both academic and commercial interest.

On the one hand, these languages are based on two (largely, but not entirely, compatible) semantics. On the one hand, **OWL DL** and **OWL Lite** semantics are based on Description Logics, which has attractive and well-understood computational properties, while **OWL Full** uses a semantic model intended to provide compatibility with RDF.

On the other hand, there is a high-level, abstract syntax for both **OWL Lite**, which is fact a subset of OWL, and **OWL DL**, a fuller style of using OWL but which places still some limitations, based on the Description Logics (DL) formalism, on how OWL ontologies are constructed. The result of eliminating these limitations is the so-called **OWL Full** language. OWL Full has the same syntax as

⁶³ <http://www.w3.org/TR/owl-ref/>; for a brief description, see also http://en.wikipedia.org/wiki/Web_Ontology_Language.

⁶⁴ It also derives from the DAML+OIL Web Ontology Language. For a brief description of both DAML and OIL, see http://en.wikipedia.org/wiki/DARPA_Agent_Markup_Language and http://en.wikipedia.org/wiki/Ontology_Inference_Layer.

RDF. The normative exchange syntax for OWL is RDF/XML, that is, OWL ontologies are most commonly expressed (implemented or, in the technical jargon, serialised) using an RDF/XML syntax.

*Since OWL is a vocabulary extension of RDF, any RDF graph constitutes an OWL Full ontology. Furthermore, the meaning associated to an RDF graph by OWL includes the meaning associated to the graph by RDF. OWL Full ontologies can thus include **any arbitrary RDF content, which is treated in a manner that is consistent with its treatment by RDF. However, OWL assigns an additional meaning to certain RDF triples.*** The OWL Semantics and Abstract Syntax document⁶⁵ shows how the RDF syntax is used in OWL and specifies exactly which triples are assigned a specific meaning (and what this meaning is).

3.3. CONCLUDING REMARKS: GUIDELINES FOR A HYBRID APPROACH TO ANNOTATION

As already mentioned, the present thesis seeks to create a joint or hybrid annotation model (that is, an annotation model which is both ontological and linguistic) suitable for the Semantic Web.

As shown in the previous sections, for an annotation model to fulfil these requirements,

1. Being suitable for the Semantic Web entails that the resulting annotations (a) must make explicit the meaning of the items being annotated; (b) must be expressed in a Semantic Web language, that is, (preferably) in XML, RDF, RDFS and/or OWL; (c) must be expressed using an ontological vocabulary (*i.e.*, an ontology-based tagset).
2. Being ontological also entails that these annotations must be encoded by means of a tagset built out of the terms of an ontology (that is, by means of the names and/or the URIs of its classes, instances and *ad hoc* relations, for example).
3. Being linguistic entails that each of these annotations must represent as well some kind of linguistic phenomenon, construct or knowledge.

Thus, the easiest and most straightforward way to achieve all of these aims seems (at least *a priori*) to base the model and its annotations in (a) an ontology or a set of ontologies that formalise(s) some kind of linguistic knowledge; (b) an annotation scheme (b.1) that can be represented in any of the Semantic Web languages aforementioned and (b.2) that uses this set of ontologies to make explicit the meaning of Web pages following a linguistic approach.

⁶⁵ <http://www.w3.org/TR/owl-semantics/>

As also shown in the previous sections, the linguistic approach to annotation is promoting as well the encoding of linguistic annotations by means of Semantic Web languages (as in ISO/LAF (2009) or in ISO/MAF (2008)). Besides, on the one hand, from a linguistic point of view, ontologies can be viewed as a way to define a formal vocabulary that expresses lexical or terminological meaning in an explicit and machine-readable way. Thus, any domain ontology could be thought of as a computational representation of the linguistic terms of that domain. Accordingly, any domain ontology could be reused within the resulting model for the annotation of the terminological information (that is, semantic information) of that domain. On the other hand, from an ontological point of view, linguistic (or lexical) ontologies, such as WordNet or EuroWordNet, together with the linguistic annotation tools that use them for sense tagging, could be reused in the context of the Semantic Web.

This is one of the advantages for the Ontological Engineering and the Semantic Web communities of uniting these areas in a hybrid approach. One of the main drawbacks of the Semantic Web is that the tools developed so far in this area for (semantic) annotation are either (1) semi-automatic, that is, they require the intervention of humans to perform or to correct (most of) their annotations, or (2) automatic but machine-based and, thus, not very accurate or not general enough. Linguistic annotation tools, on the contrary, are automatic, more general, and can be more accurate than most automatic Semantic Web annotation tools. Therefore, retargeting linguistic annotation tools, translating their annotated outputs to a Semantic Web language and mapping their linguistic tags to an ontology-based tagset should provide the Semantic Web with the automatic and robust annotation tools that it lacks.

As for the advantages for the Linguistic Annotation community of this hybrid approach, basically, it helps linguistic annotation tools and annotations interoperate, that is, it helps compare, merge, combine and integrate them, as shown in the development of the present work. Indeed, one of the main objectives of this work was to integrate and combine the results of several linguistic tools to perform the hybrid annotations associated to this model. However, as a rule, each linguistic annotation tool encodes differently the linguistic phenomena that annotates. Therefore, its annotations cannot interoperate that easily with the results of any other linguistic annotation tool. On the contrary, they require a previous mapping to a common language. Ontologies were found the ideal way to express in a common language the annotations that had to interoperate. Once they were expressed in a common (ontology-based) language, they could interoperate rather straightforwardly. The following chapters show how this was achieved in OntoTag and in OntoTagger.

4. ONTOTAG: THE HYBRID ANNOTATION MODEL

As mentioned in the previous chapters, linguistic annotation tools have proved to be very helpful to reduce drastically the amount of time and efforts required to annotate text, which is indeed a very high time-consuming task. However, linguistic annotation tools have still some limitations

1. They usually introduce a certain rate of errors and ambiguities when tagging. This error rate can range from 10 percent up to 50 percent of the units annotated for unrestricted, general texts.
2. They cover only some levels of linguistic description (usually just one of them) in their annotations for a particular language.
3. The tagsets, the schemas and the conventions adopted when these tools were being developed were usually determined *ad hoc* (and, sometimes, even on the fly). Accordingly, most often, they do not comply with the standards and guidelines existing nowadays for the level(s) at which each one of them operates and tags.

OntoTag tries to solve each of these limitations, by assembling and wrapping several linguistic tools into the same architecture. As commented before, the aim of this annotation architecture is to show that it is possible to:

1. Reduce automatically the rate of errors introduced by a linguistic annotation tool for a given level when tagging, at least at the morphosyntactic level. This can be achieved by contrasting and combining its results with the ones coming from another (or other) linguistic annotation tool(s) operating at the same level.
2. Make a number of linguistic annotation tools interoperate and, hence, obtain a final annotation by (a) summing up the results of different linguistic annotation tools tagging at the same level (the morphosyntactic level, in this case), (b) interconnecting the results of different linguistic annotation tools tagging at different levels or layers, or (c) pipelining the output of a given (set of) linguistic annotation tools, as input, to another (other) tool(s) tagging at a higher-level (from a linguistic point of view).
3. Map⁶⁶ tool-dependent annotations onto a sort of tool-independent annotations and standardise them. As a spin-off, this process of standardisation is expected to (a) facilitate the interoperation of all the tools being assembled and (b) make the architecture more modular.

In order to solve these limitations, this model details both an abstract annotation architecture and an abstract annotation scheme for the linguistic annotation of documents (mostly web pages) at different

⁶⁶ In its mathematical sense, that is, “to make a correspondence”.

levels. Both elements are supported and linked by the third pillar of OntoTag, its set of (linguistic) ontologies.

The abstract annotation architecture of OntoTag aims at (a) enabling the integration and interoperability of several linguistic annotation tools operating at the same or different levels of linguistic description; (b) comparing and combining their results for their improvement and interoperability.

The goal of **the abstract annotation scheme of OntoTag** is showing (i) how the annotations of linguistic tools operating at different linguistic levels or layers can be summed up and interconnected; and (ii) how tool-dependent linguistic annotations can be mapped onto a sort of tool-independent annotations and, in this way, be standardised.

The linguistic knowledge required for representing, mapping, standardising, comparing, combining, summing up, interconnecting and linking linguistic annotations by means of OntoTag is portrayed in **OntoTag's linguistic ontologies** (see Figure 6 on page 73). These ontologies formalise the elements and relationships pertaining to the main levels of linguistic description (namely the morphosyntactic, the syntactic and the semantic levels). This formalisation was obtained according to existing annotation recommendations and guidelines, when any existed for that precise level. For example, the EAGLES recommendations for morphosyntactic (EAGLES, 1996a) and syntactic (EAGLES, 1996b) annotation were taken into account and formalised conveniently when developing these ontologies.

Since the OntoTag model had to be suitable for the Semantic Web, its output annotations had to comply with the previous requirement. This is the main reason why OntoTag's ontologies were developed. Indeed, linguistic annotation tools can make explicit the grammatical meaning and even the semantics of texts by means of their annotations, but these annotations cannot be interpreted currently by machines in the same way as semantically annotated Semantic Web documents. Actually, Semantic Web contents are annotated with ontological terms. Therefore, linguistic annotations have to be transformed beforehand into ontological annotations⁶⁷ to be processed as Semantic Web annotations. After such a transformation, the meaning extracted by linguistic annotation tools, previously only human-readable, is made explicit for machines to read and/or interpret it. This is enabled in OntoTag both (a) by the formalisation of the knowledge associated to linguistic annotations included in OntoTag's ontologies and (b) by mapping the results of linguistic annotation tools onto ontological annotations based on these ontologies.

⁶⁷ The terms **ontological annotation** and **ontology-based annotation** refer to an annotation whose tagset consists of terms extracted from one or more ontologies.

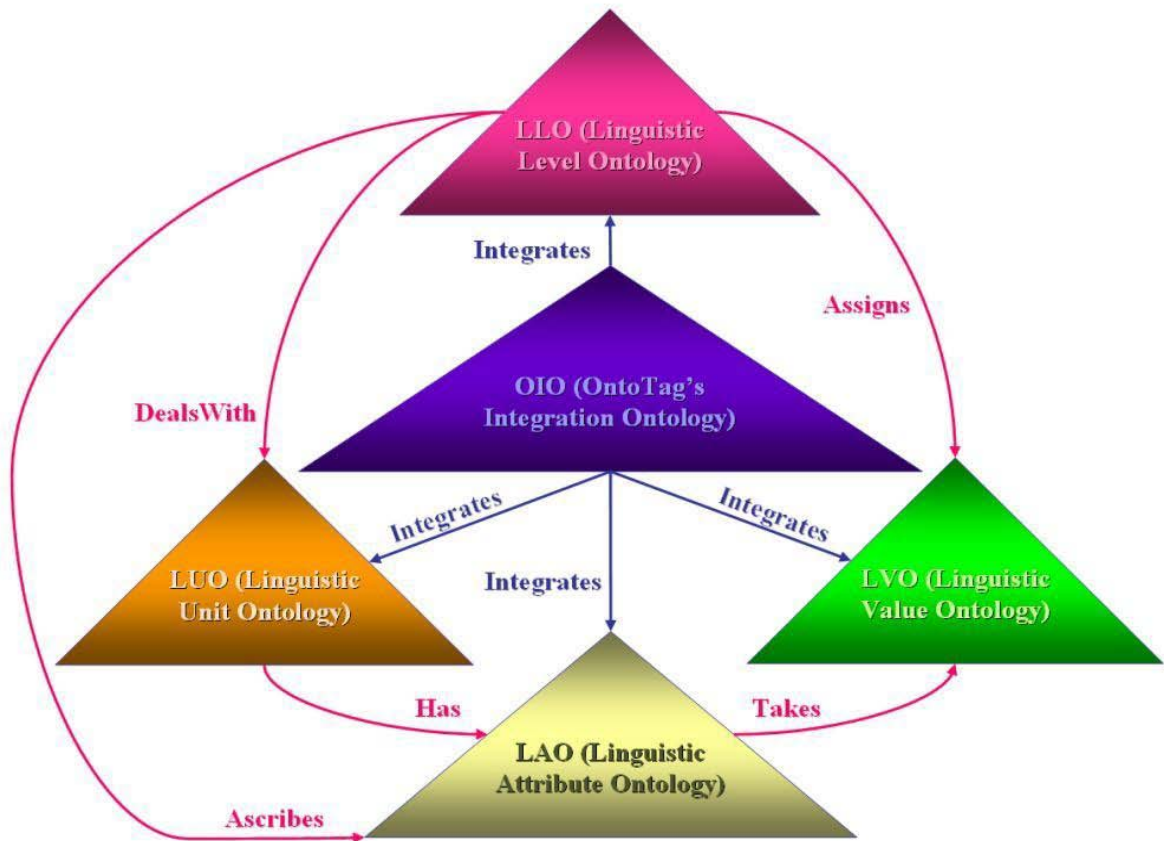


Figure 6: OntoTag's Linguistic Ontologies

Thus, briefly, OntoTag's ontologies are used as the central resource in this model for capturing and representing the meaning of its inputs, that is, natural language texts. This capture and representation of meaning follows the Ontological Semantics theory of Niremburg & Raskin (2004). That is, meaning is made explicit by means of a linguistically-motivated ontological annotation or, equivalently, by means of an ontology-based linguistic annotation. Consequently, the outputs produced by OntoTag's implementations should be suitable for the Semantic Web (and its different purposes). A general view of the approach followed when developing the OntoTag model is shown in Figure 7 (on page 74).

Accordingly, the rest of this chapter has been structured as follows: first, the different OntoTag's ontologies are presented in Section 4.1, each one in a dedicated subsection. Then, OntoTag's (abstract) annotation architecture will be presented in Section 4.2. Finally, OntoTag's (abstract) annotation scheme will be presented in Section 4.3.

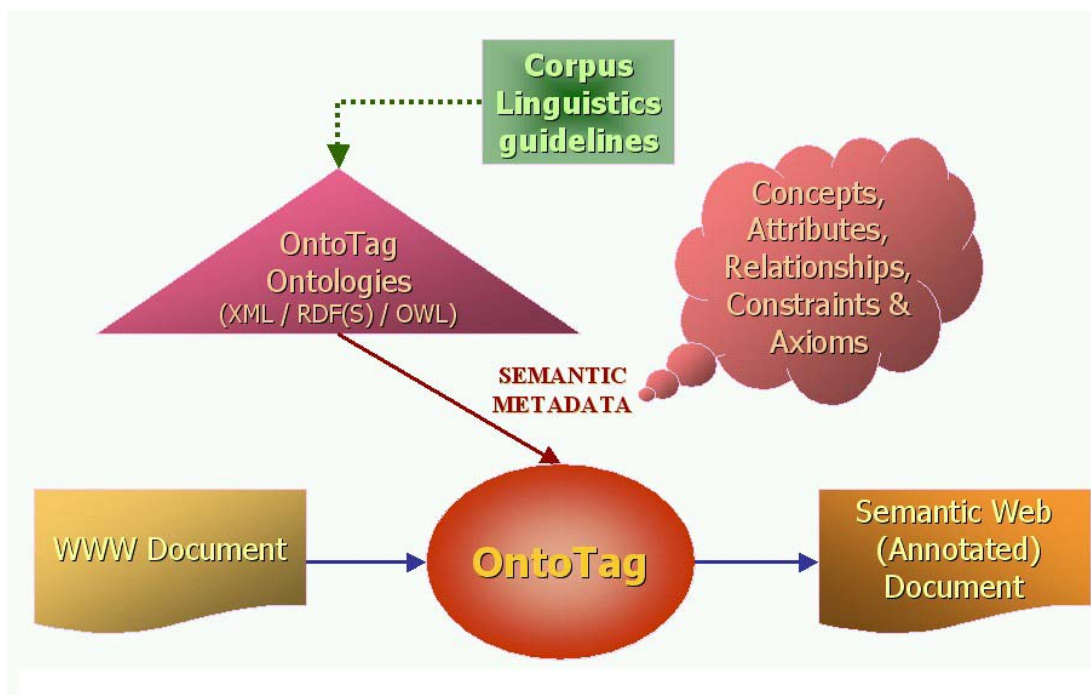


Figure 7: OntoTag: a Hybrid (Linguistic and Ontological) Annotation Model for the Semantic Web

4.1. ONTOTAG'S LINGUISTIC ONTOLOGIES

As mentioned above, one of the main components of the OntoTag model is its set of linguistic ontologies⁶⁸, devised to formalise the structure and relationships holding between the elements of language at different linguistic levels. The kind of elements and relationships considered in these ontologies are (1) the ones usually included in existing annotation schemas, such as morphosyntactic units (EAGLES, 1996a) and dependency relations (Tapanainen & Järvinen, 1997); and (2) those others needed to link the first ones together.

Thus, first, a Linguistic Level Ontology (LLO) was developed both to capture the stratification of natural language analysis and generation and to simplify the formalization of the other elements. Up to date, it is a very simple and small ontology, containing only the concepts Linguistic Level, Morphological Level, Syntactic Level, Semantic Level, Discourse Level and Pragmatic Level. It clearly needs to be further expanded, but this expansion fell out of the scope of this thesis and was left for further research. This is also the reason why it is not presented in more detail here.

⁶⁸ For a proper definition of the term 'ontology', in the sense it is used here, see Borst (1997), Gruber (1993) and also Studer *et al.* (1998).

Second, following the EAGLES (1996a) guidelines for morphosyntactic annotation of corpora, but obviously broadening its scope, three different ontologies were built to represent the <Category, Attribute, Value> Triple formalism at all levels of annotation: a Linguistic Unit Ontology (LUO), a Linguistic Attribute Ontology (LAO), and a Linguistic Value Ontology (LVO). These three ontologies (LUO, LAO and LVO) are interconnected by *ad hoc* relations such as, for example, *Has*(Linguistic Unit, Linguistic Attribute), or *Takes*(Linguistic Value, Linguistic Attribute). The levels formalised are the morphosyntactic, the syntactic and the semantic.

And third, the OntoTag Integration Ontology (OIO) establishes the main interrelations between documents (annotated and non-annotated), levels, units, attributes and values, both in the linguistic and in the ontological areas of annotation. Each of these ontologies is described in a dedicated subsection (from Subsection 4.1.2 to Subsection 4.1.5).

The **application of these five ontologies within the OntoTag annotation model** is threefold: (1) as discussed above, they identify the different elements (mostly linguistic, but also ontological) that can appear in linguistic annotations; (2) they are applied to transforming linguistic annotations into ontological annotations, suitable for the Semantic Web; and (3) once the ontologies have been populated (or instantiated) by the annotations obtained with OntoTag implementations, they can also act as a repository or database of these annotations.

All of them were developed using the WebODE⁶⁹ workbench and the METHONTOLOGY methodology (Gómez-Pérez *et al.*, 2004). This methodology has to be presented before describing the ontologies mentioned. Hence, an introduction to METHONTOLOGY has been included in the following subsection (4.1.1).

Lastly, all of them were included, evaluated and validated in the first implementation of OntoTag, *i.e.*, OntoTagger. The results of their evaluation and validation are shown in Chapter 6 (Results and Evaluation) and in Chapter 7 (Conclusions).

4.1.1. BUILDING ONTOLOGIES WITH METHONTOLOGY

METHONTOLOGY (Gómez-Pérez *et al.*, 2004) ‘proposes an ontology building life cycle based on *evolving prototypes* because it allows adding, changing, and removing terms in each new version (prototype)’. In this evolving prototype life cycle, it is proposed as well to reuse pre-existing

⁶⁹ <http://webode.dia.fi.upm.es/>.

ontologies whenever possible, after evaluating and adapting them when needed. If no ontology can be reused, a new one has to be created.

The components eventually included in an ontology developed by means of METHONTOLOGY are the following:

Concepts (which are taken in a broad sense) are the basic elements or entities of the knowledge domain or area being formalised in the ontology. For instance, in the linguistic domain, `Token`, `Verb`, `Sentence`, etc. are concepts. The concepts of an ontology are usually organised in taxonomies through which inheritance mechanisms can be applied. For instance, a taxonomy of linguistic units (or concepts) can be built, where a `Verb` is a type of a `Morpho-Syntactic Unit`, a `Clause` is a type of `Syntactic Unit`, etc.

Relations represent a type of association between concepts of the domain. If the relation links two concepts, for example, *Realises*, which links a `Term` to a `Linguistic Unit`, it is called a binary relation. Each binary relation may have an inverse relation that links the concepts in the opposite direction. There are specific relations, which help arrange the concepts of the ontology into one or more *concept taxonomies*⁷⁰ and/or *meronomies*⁷¹. Taxonomies are essentially built out of the **Subclass-Of** relation: a concept C_1 is a **Subclass-Of** another concept C_2 if and only if every instance of C_1 is also an instance of C_2 . However, special attention must be paid to the identification of sets of (i) disjoint concepts in a conceptual taxonomy, that is, concepts that cannot have common instances; and (ii) concepts that account for all the (possible) instances of another, more general concept, that is, exhaustive decompositions of other concepts. Thus, three other taxonomic relations are proposed in METHONTOLOGY (Gómez-Pérez *et al.*, 2004), namely *Disjoint-Decomposition*, *Exhaustive-Decomposition* and *Partition*:

- A **Disjoint-Decomposition** of a concept C is a set of subclasses of C that do not share common instances and may not cover C , that is, there can be instances of C that are not instances of any of the concepts in the decomposition.
- An **Exhaustive-Decomposition** of a concept C is a set of subclasses of C that cover C and may have common instances or subclasses, that is, there cannot be instances of C that are not instances of at least one of the concepts in the decomposition.
- A **Partition** of a concept C is a set of subclasses of C that does not share common instances and that covers C , that is, every instance of C is an instance of one (and only one) of the concepts in

⁷⁰ **Taxonomies** are built out of the well-known *IsA* relation, which associates an entity (called the *hyperonym* concept) of a certain type to another entity (called the *hyponym* concept) of a more general type. Taxonomies, thus, capture the type/subtype relations of a domain (EAGLES, 1999).

⁷¹ **Meronomies** describe the *Part-Whole* (or *Part-Of*) relations that hold between the concepts of the domain being modelled (EAGLES, 1999).

the partition. Partitions cannot be represented at the moment in WebODE, the environment used to develop OntoTag’s ontologies and, hence, this type of taxonomic relation has been modelled by means of a *Disjoint-Decomposition*, putting the emphasis on the fact that the subclasses that constitute the decomposition do not share any of their instances.

Instances are used to represent elements or individuals in an ontology. For example, in the OIO, (1) FDG is an instance of the concept `Linguistic Annotation Tool`; (2) `AI.1234` could be an instance of the concept `Annotation`; and (3) the Spanish word ‘amigo’ could be an instance of the concept `Linguistic Unit`. Relations can be instantiated too, *e.g.*, it can be said that `FDG performs AI.1234` and that ‘amigo’ is annotated with `AI.1234`.

Constants are numeric values that do not change at all, or at least for a long time as, for instance, the number `PI = 3.1416`.

Attributes describe the properties of instances and of concepts. Two different types of attributes can be distinguished: instance attributes and class attributes. **Instance attributes** describe concept instances, where they take their values. These attributes are defined in a concept and inherited by its sub-concepts and instances. For example, `author` is an instance attribute of the concept `Document` in the LUO, since its value would be proper to each instance of this concept (that is, each `Document` has its own `author(s)`). **Class attributes** describe concepts and take their values in the concept where they are defined. Class attributes are neither inherited by the subclasses nor by the instances. The attribute `GrammaticalCategory` of `Morpho-Syntactic Unit` is an example of a class attribute, since all nouns share the same `GrammaticalCategory` (*i.e.*, `Noun`), all verbs share the same `GrammaticalCategory` (`Verb`), etc. On the one hand, ontology development tools usually provide predefined domain-independent class attributes for all the concepts, such as the concept *description*, its *synonyms*, its *acronyms*, etc. On the other hand, the users of these tools must define the domain-dependent class attributes of the ontologies they develop.

Formal axioms are logical expressions that are always true and are normally used to specify constraints in the ontology. An example of axiom is: “The `Linguistic Values` that the `Gender Linguistic Attribute` can take **are restricted to** `Masculine`, `Feminine`, `Neuter` and `Common in Danish`”.

Finally, **rules** are expressions, usually specified using the template *if* <conditions> *then* <consequent>, that are generally used to infer knowledge in the ontology, such as attribute values, relation instances, etc. For example, the following rule

if (Language=German) \wedge Category (Token, Common Noun) \wedge Ends (Token, Lemma, '-tion')
then Value (Token, Gender, Feminine)

fixes the Linguistic Value (Feminine) of a Linguistic Attribute (Gender) for those German common nouns ending with '-tion'.

METHONTOLOGY also proposes to conceptualise each new ontology by means of a set of tabular and graphical intermediate representations. These intermediate representations are obtained after each of the eleven tasks for ontology conceptualisation included in this methodology has been completed. Such intermediate representations allow modelling the different components described above.

The eleven tasks of METHONTOLOGY are the following: (1) building the glossary of terms of the (sub-)domain being conceptualised into the ontology; (2) building the taxonomy (or subclassification) of the concepts of this (sub-)domain; (3) identifying the *ad hoc* (binary) relations that hold between these concepts; (4) creating the concept dictionary, which includes, for each concept, its instances, its class and instance attributes, and its *ad hoc* relations; describing in detail (5) each *ad hoc* relation, (6) each instance attribute, (7) each class attribute, and (8) each constant in the concept dictionary; describing (9) the formal axioms and (10) the rules that hold for the concepts in the (sub-)domain in question; and optionally (11) introducing in the ontology the information about instances.

The different ontologies of the OntoTag model are described below. As already mentioned, they have been developed within WebODE, following the METHONTOLOGY methodology. However, only the intermediate representations obtained in the conceptualisation of the OIO have been included here for the sake of space. As for the remaining ontologies, just the components resulting from the application of METHONTOLOGY for each of them (namely their concepts, relations, instances, constants, attributes, formal axioms and rules) are discussed in the following subsections.

4.1.2. THE ONTOTAG INTEGRATION ONTOLOGY (OIO)

The OntoTag Integration Ontology (OIO) constitutes a higher-level, comprehensive conceptualisation of the elements of the OntoTag model and the relationships that hold between them. The concepts of this conceptualisation are described and subclassified in the rest of OntoTag's ontologies. Besides, this ontology establishes the key metadata used in OntoTag's annotations. Therefore, this ontology constitutes, in fact, a sort of knowledge representation ontology for the domains of ontological and linguistic annotation.

A detailed description of this ontology is presented in the next subsections, following the METHONTOLOGY (Gómez-Pérez *et al.*, 2004) methodological steps for ontology development from scratch. Hence, (i) the glossary of terms of the OIO is presented in Subsection 4.1.2.1; (ii) its concept

taxonomies, in Subsection 4.1.2.2; (iii) its *ad hoc* relations, in Subsection 4.1.2.3; (iv) its concept dictionary, in Subsection 4.1.2.4; (v) its detailed tables, in Subsection 4.1.2.5 (respectively, the OIO *ad hoc* binary relation table in Subsection 4.1.2.5.1, the OIO instance attribute table in Subsection 4.1.2.5.2, the OIO class attribute table in Subsection 4.1.2.5.3, and the OIO constant table in Subsection 4.1.2.5.4); (vi) its formal axiom and rule tables, in Subsection 4.1.2.6; (vii) its instance table, in Subsection 4.1.2.7; and, finally, (viii) some statistics about the number of terms (*i.e.*, components) of the OIO are presented in Subsection 4.1.2.8.

4.1.2.1 OIO GLOSSARY OF TERMS

The **glossary of terms** of an ontology identifies the set of all the relevant terms of the domain being conceptualised, and accompanies them with their natural language definitions, their synonyms and acronyms, as well as the type of term they belong to, namely, concept, group, class attribute⁷², instance attribute, value, instance, relation, constant, etc. The OIO glossary of terms has been distributed (for the sake of readability) into Table 9 (concepts), Table 10 (groups), Table 11 (attributes), Table 12 (*ad hoc* relations) and Table 13 (instances).

Table 9: OIO glossary of terms (extracted from WebODE) – concepts

Name	Synonyms	Acronyms	Description	Type
Annotation	Mark-Up Tag Metadata Label	--	This concept formalises mainly the different types of information that can be attached to a text <code>Unit</code> either (i) to gloss it, (ii) to state its format of presentation, or (iii) to make its meaning explicit. It also incorporates the properties associated to the concept (entity) that results from the decomposition of the ternary <i>ad hoc</i> relation <i>Annotates</i> (<i>Annotator</i> , <i>Content</i> , <i>Annotation</i>), which cannot be represented as such within the underlying model of the ontology development tool WebODE (only binary <i>ad hoc</i> relations can be represented in WebODE).	Concept
Annotation Tool	Mark-Up Tool Tagger	--	This concept denotes all kinds of machine programs used to attach some metadata to a text <code>Unit</code> or a set of text units (in a particular language).	Concept
Annotator	Tagger	--	An <i>Annotator</i> is a person or a program that tags, annotates or marks up documents or content(s).	
Applet	--	--	An <i>Applet</i> is any small application that performs one specific task (Wikipedia, 2009). However, this term typically refers to those programs written in Java that are included in an HTML page.	Concept

⁷² There are two different types of attributes: instance attributes and class attributes. **Instance attributes** are those attributes whose value(s) may be different for each instance of the concept, whilst **class attributes** describe concepts and take their values in the class where they are defined.

Name	Synonyms	Acronyms	Description	Type
Attribute	Feature Property Quality Characteristic	--	According to Wikipedia (2009), an Attribute is (a) in Philosophy, a property, an abstraction of a characteristic of an entity or substance; (b) in Computing, a specification that defines a property of an object, element, or file. In our case, the term Attribute refers to a property of a text Unit that (i) can take some possible values and (ii) can be <i>Part-Of</i> a Triple Unit-Attribute-Value.	Concept
Content	--	--	Content can mean any creative work, such as text, graphics, images or video (Wikipedia, 2009).	Concept
Corpus	Text Corpus	--	Following Wikipedia (2009) and EMELD (2009), a (linguistic) Corpus is a collection of texts (now usually in an electronic format) that are chosen and organized so as to facilitate linguistic research, <i>e.g.</i> , to represent a certain type of discourse or to provide a data set to be searched for examples of linguistic features (http://emeld.org/school/glossary.html#corpus).	Concept
Document	File	--	A Document is a bounded physical representation of a body of information designed with the capacity (and usually intent) to communicate (Wikipedia, 2009).	Concept
Formal Language	Artificial Language	--	Following Wikipedia (2009), a Formal Language is a set of words, <i>i.e.</i> , finite strings of letters, symbols, or tokens. The set from which these letters are taken is called the alphabet over which the language is defined. A formal language is often defined by means of a formal grammar (also called its formation rules). Formal languages are studied in Computer Science and Linguistics.	Concept
Human Annotator	--	--	A person who performs annotations of any type, either manually or with an annotation editing or a computer-aided annotation tool.	Concept
Hybrid Annotation	--	--	A Hybrid Annotation is an Annotation that is a Linguistic Annotation, an Ontological Annotation and a Unit-Attribute-Value Triple at a time.	Concept
Hybrid Annotation Tool	--	--	The Hybrid Annotation Tool concept represents all the different annotation tools which are an <i>Instance-Of</i> Linguistic Annotation Tool and Ontological Annotation Tool (at a time). In particular, this includes those linguistic annotation tools whose annotations are formulated by means of a linguistically-motivated Ontological Tagset.	Concept
Hybrid Tagset	--	--	A Hybrid Tagset is a Tagset that is both a Linguistic Tagset and an Ontological Term Set (at a time). In particular, this includes all those ontological term sets that result from the formalisation in an Ontology of a given Linguistic Tagset.	Concept
Hybridly Annotated Document	Hybridly Marked-Up Document Hybridly Tagged Document	--	A Hybridly Annotated Document is a Marked-Up Document that is both linguistically and ontologically annotated.	Concept

Name	Synonyms	Acronyms	Description	Type
Image	Pic Picture Figure	--	An Image (Wikipedia, 2009) is an artefact (for example, a two-dimensional picture) that has a similar appearance to some subject—usually a physical object or a person. The word Image is also used in a broader sense to refer to any two-dimensional figure such as a map, a graph, a pie chart, or an abstract painting. As for the present ontology, this concept denotes all kinds and formats (<i>i.e.</i> , JPEG, GIF, etc.) of (digital) images that are usually included in web documents.	Concept
Language	--	--	Following Wikipedia (2009), a Language is a particular kind of system for encoding and decoding information. This necessarily involves the systematic creation and usage of systems of symbols – each symbol referring to linguistic concepts with semantic or logical or otherwise expressive meanings.	Concept
Linguistic Annotation	--	--	A Linguistic Annotation is a type of Annotation that is linguistically-motivated, that is, which results from the application of a Linguistic Tagset.	Concept
Linguistic Annotation Layer	--	--	This concept formalises the second step towards the stratification (or modularisation) of a Linguistic Level for its annotation. Each Linguistic Annotation Layer is in charge of the annotation of some interrelated linguistic phenomena of a given Linguistic Level. Thus, a Linguistic Annotation Layer results from the aggregation of the different units, attributes, values and relations that formalise these linguistic phenomena, together with their respective (types of) annotations.	Concept
Linguistic Annotation Tool	--	--	This concept represents all the different types of annotation tools that perform linguistic annotations.	Concept
Linguistic Attribute	--	--	This concept formalises the different kinds of linguistically-motivated attributes that can be included in any kind of Linguistic Annotation or, equivalently, in any kind of linguistically-motivated Triple Unit-Attribute-Value.	Concept
Linguistic Level	--	--	This concept depicts the main different points of view (levels) under which linguistic units are studied and annotated, such as Morphology, Syntax, Semantics, etc.	Concept
Linguistic Tagset	--	--	A Linguistic Tagset is a Tagset designed according to linguistic criteria, theories and/or formalisms.	Concept
Linguistic Unit	--	--	This concept represents all the different kinds of linguistically-motivated units that can be included in any kind of Linguistic Annotation or, equivalently, in any kind of linguistically-motivated Triple Unit-Attribute-Value. An alternative definition can be found in Downing and Locke (1992). According to these authors, a Linguistic Unit can be defined as ‘any stretch of language which constitutes a semantic whole and which has a recognised pattern that is repeated regularly in speech and writing’.	Concept
Linguistic Value	--	--	The Linguistic Value concept represents all the possible types of linguistically-motivated values that can be included in any kind of Linguistic Annotation or, equivalently, in any kind of linguistically-motivated Triple Unit-Attribute-Value.	Concept

Name	Synonyms	Acronyms	Description	Type
Linguistically Annotated Document	Linguistically Marked-Up Document Linguistically Tagged Document	--	A Linguistically Annotated Document is a type of Annotated Document that contains a set of linguistic annotations, performed according to some linguistic criteria, theories or formalisms and/or using a certain Linguistic Tagset.	Concept
Marked-Up Document	Annotated Document Tagged Document	--	A Marked-Up Document is a Document that contains some kind of Annotation attached to (some part of) its Content.	Concept
Natural Language	--	--	According to Wikipedia (2009), in the philosophy of language, a Natural Language is any language which arises in an unpremeditated fashion as the result of the innate facility for language possessed by the human intellect. A Natural Language is typically used for communication, and may be spoken, signed, or written.	Concept
Non-Marked-Up Document	Non-Annotated Document Untagged Document	--	A Non-Marked-Up Document is a Document that contains no Annotation attached to (any part of) its Content.	Concept
Ontological Annotation	Ontology-Based Annotation	--	An Ontological Annotation is a type of Annotation that is ontology-based, that is, which results from the application of an Ontological Tagset and/or some ontological criteria or formalism.	Concept
Ontological Annotation Tool	--	--	An Ontological Annotation Tool is an Annotation Tool that performs ontological annotations.	Concept
Ontological Tagset	Ontology-Based Tagset	--	An Ontological Tagset is a Tagset that consists of some of (or all) the terms of one or more ontologies, either as such or (slightly) modified.	Concept
Ontological Term	Ontological Component	--	This concept represents all the different entries that can appear in the METHONLOGY-derived term glossary of a particular Ontology. More specifically, all the names (or the identifiers) of the different components that can be included in an Ontology are <i>Instances-Of</i> this concept.	Concept
Ontological Term Set	--	--	The Ontological Term Set is a set that comprises the different terms of an Ontology. In other words, an Ontological Term Set is a set that contains the names, the synonyms and the acronyms of the components included in the term glossary of an Ontology. In particular, this includes the names, synonyms and acronyms of its concepts, attributes and <i>ad hoc</i> relations.	Concept
Ontologically Annotated Document	Ontologically Marked-Up Document Ontologically Tagged Document	--	An Ontologically Annotated Document is a type of Annotated Document that contains a set of ontological annotations, performed according to some ontological criteria or formalisms and/or using a certain Ontological Tagset.	Concept
Ontology	--	--	According to Gruber (1993), Borst (1997) and Studer et al. (1998), an Ontology is a formal (and explicit) specification of a shared conceptualisation.	Concept
OntoTag-Annotated Document	OntoTag-Marked-Up Document OntoTag-Tagged Document	--	The concept OntoTag-Annotated Document represents all the different types of documents (final and intermediate) marked up following the OntoTag annotation model, using a particular Tagset, a particular mark-up Formal Language and a particular Ontology or set of ontologies.	Concept

Name	Synonyms	Acronyms	Description	Type
Other Annotation	--	--	This concept comprises the set of those <i>instances of</i> Annotation that cannot be classified as any of the remaining classes into which the concept Annotation has been <i>exhaustively-decomposed</i> , namely Linguistic Annotation, Ontological Annotation and Subject-Predicate-Object Triple.	Concept
Other Attribute	--	--	The <i>instances of</i> this concept are those attributes that cannot be considered a Linguistic Attribute (or any of its subclasses, had it any).	Concept
Other Content	--	--	This class comprises all the <i>instances of</i> Content (i) that are usually inserted in a Web Document but (ii) which cannot be classified as <i>instances of</i> any of the types of Content explicitly formalised in the ontology (namely Plain Text, Image and Applet). However, no <i>Instance-Of</i> Other Concept was found in the ODECorpus-Cinema.	Concept
Other Marked-Up Document	Other Annotated Document Other Tagged Document	--	The <i>instances of</i> this concept are those <i>instances of</i> Marked-Up Document that cannot be classified as a pure Linguistically Annotated Document and/or an Ontologically Annotated Document (or any of their subclasses).	Concept
Other Tagset	--	--	Any Tagset that cannot be subclassified as a pure Linguistic Tagset and/or a pure Ontological Tagset (or any of their subclasses) is an <i>Instance-Of</i> this concept.	Concept
Other Unit	--	--	This concept represents all those <i>instances of</i> Unit that cannot be classified as pure linguistic units (or any of the subclasses of Linguistic Unit, had it any).	Concept
Other Value	--	--	The <i>instances of</i> this concept are those values that cannot be considered a Linguistic Value (or any of its subclasses, had it any).	Concept
Plain Text	--	--	This concept represents the textual information that remains after extracting all other Content from a Web Document (<i>i.e.</i> , images, applets, labels, etc.) and that is to be annotated following the OntoTag model. The values of the instance attributes of this concept are extracted and assigned from the original mark-up of the Web Document for further processing.	Concept
Subject-Predicate-Object Triple	Triple Subject-Predicate-Object	SPO Triple	As with RDF triples, a Subject-Predicate-Object Triple is the aggregation of three components: (i) a subject, (ii) a predicate and (iii) an object. In such a triple, the object is the value of the attribute or the property specified by the predicate for the subject of the triple.	Concept
Tagset	--	--	A Tagset is a set of tags used to annotate a particular Corpus (<i>i.e.</i> , a particular set of documents) in a particular language (http://emeld.org/school/glossary.html#tagset).	Concept
Unit	--	--	The concept Unit stands for the essentials of item study and classification, that is, the object of study of a particular domain or sub-domain. This concept constitutes the main (first) element of a Triple Unit-Attribute-Value that constitutes an Annotation for an item.	Concept
Unit-Attribute-Value Triple	Triple Unit-Attribute-Value	UAV Triple	This concept represents all the possible associations (Unit, Attribute, Value) that configure a possible Annotation for a text Unit.	Concept

Name	Synonyms	Acronyms	Description	Type
Value	--	--	The concept Value represents all those elements of a domain that can be assigned to some Attribute ascribable to some Unit of that domain, that is, the elements that can function as the third element of a Unit-Attribute-Value Triple.	Concept
Web Document	Web Page	--	This concept formalises the different types of digital (possibly HTML marked-up) documents that can be retrieved from the World Wide Web (WWW).	Concept

Table 10: OIO glossary of terms (extracted from WebODE) – groups

Name	Synonyms	Acronyms	Description	Type
Annotation Group	--	--	--	Group
Annotation Tool Group	--	--	--	
Annotator Group	--	--	--	
Attribute Group	--	--	--	
Content Group	--	--	--	
Document Group	--	--	--	
Language Group	--	--	--	
Marked-Up Document Group	--	--	--	
Tagset Group	--	--	--	
Unit Group	--	--	--	
Value Group	--	--	--	

Table 11: OIO glossary of terms (extracted from WebODE) – attributes

Name	Synonyms	Acronyms	Description	Type
<i>Annotation Tool:</i> <i>isHuman</i> <i>Human Annotator:</i> <i>isHuman</i>	--	--	The <i>isHuman</i> (class) attribute tells human annotators from annotation tools.	Class Attribute
<i>Formal Language:</i> <i>isNatural</i> <i>Natural Language:</i> <i>isNatural</i>	--	--	This class attribute differentiates natural languages from formal (or artificial) languages.	Class Attribute
<i>Hybrid Annotation:</i> <i>conceptType</i> <i>Hybrid Annotation Tool:</i> <i>conceptType</i> <i>Hybrid Tagset:</i> <i>conceptType</i> <i>Linguistic Annotation:</i> <i>conceptType</i> <i>Linguistic Annotation Tool:</i> <i>conceptType</i> <i>Linguistic Attribute:</i> <i>conceptType</i> <i>Linguistic Tagset:</i> <i>conceptType</i> <i>Linguistic Unit:</i> <i>conceptType</i> <i>Linguistic Value:</i> <i>conceptType</i> <i>Ontological Annotation:</i> <i>conceptType</i> <i>Ontological Annotation Tool:</i> <i>conceptType</i> <i>Ontological Tagset:</i>	--	--	Most of the subclassifications included in this ontology have been developed in terms of the nature or the domain associated to the concepts in question, <i>i.e.</i> , whether they are LINGUISTIC, ONTOLOGICAL, HYBRID (both LINGUISTIC and ONTOLOGICAL) or none of them (OTHER). The <i>conceptType</i> class attribute fixes the nature of the concept it is attached to, according to the values mentioned.	Class Attribute

Name	Synonyms	Acronyms	Description	Type
conceptType				
alternativeText	--	--	Some browsers, even though they may interpret the (HTML) tag 'applet', cannot execute the corresponding Applet (since a suitable Java plug-in has not been installed yet, for example). In this case, the alternativeText carried by this attribute is displayed by the browser.	Instance Attribute
annotationID	--	--	The decomposition of the ternary <i>ad hoc</i> relation <i>Annotates</i> (Annotator, Content, Annotation), mentioned in the description of the Annotation concept, entails the addition of this attribute to this concept (for integrity reasons), together with the attributes annotatorID and contentID.	Instance Attribute
annotatorID	--	--	The decomposition of the ternary <i>ad hoc</i> relation <i>Annotates</i> (Annotator, Content, Annotation), mentioned in the description of the Annotation concept, entails the addition of this attribute to this concept (for integrity reasons), together with the attributes annotationID and contentID.	Instance Attribute
appletCode	--	--	This attribute designates the (Java) 'class' or 'jar' File to be run when an Applet is invoked.	Instance Attribute
appletCodeBase	--	--	The locations of the code and the rest of the elements associated to an Applet are captured in this attribute by means of a Universal Resource Locator (URL), which details where they can be found.	Instance Attribute
appletHeight	--	--	The appletHeight attribute determines the height (on the screen) of the frame in which an Applet will be executed.	Instance Attribute
appletName	--	--	This attribute identifies univocally an Applet when it needs to communicate with some other Applet(s) or element(s) which are being run within the same Web Page or frame.	Instance Attribute
appletWidth	--	--	The appletWidth attribute determines the width (on the screen) of the frame in which an Applet will be executed.	Instance Attribute
attributeID	--	--	In order to have perfectly fixed the particular Unit, Attribute and Value that constitute a particular Unit-Attribute-Value Triple, they are linked together by means of their corresponding identifiers, namely unitID, attributeID, valueID and tripleID.	Instance Attribute
author	creator	--	This attribute of Document identifies the creator of the Document in question, be it human or a computer tool.	Instance Attribute
backgroundColour	--	--	This attribute identifies the background colour with which a (portion of) Plain Text is displayed in the Document it comes from.	Instance Attribute
contentID	--	--	The decomposition of the ternary <i>ad hoc</i> relation <i>Annotates</i> (Annotator, Content, Annotation), mentioned in the description of the Annotation concept, entails the addition of this attribute to this concept (for integrity reasons), together with the attributes annotatorID and annotationID.	Instance Attribute
creationDate	--	--	The creationDate attribute keeps track of the date of creation of a Document.	Instance Attribute
documentURL	--	--	The location of each Document is kept in this attribute, which stores its URL (or URLs, if the	Instance Attribute

Name	Synonyms	Acronyms	Description	Type
			Document is replicated in multiple locations).	
fontColour	--	--	This attribute identifies the colour of the characters (fonts) with which a (portion of) Plain Text is displayed in the Document it comes from.	Instance Attribute
fontSize	--	--	The fontSize attribute stores the information about the height and width (size) of the characters (fonts), with which a (portion of) Plain Text is displayed in the Document it comes from.	Instance Attribute
fontType	--	--	This attribute keeps track of the kind of font (Arial, Courier, Times New Roman, etc.) with which a (portion of) Plain Text is displayed in the Document it comes from.	Instance Attribute
imageFormat	--	--	The information about the type of encoding of an Image (GIF, JPEG, TIFF, etc.) is stored in this attribute.	Instance Attribute
imageHeight	vertical Precision	--	The imageHeight attribute of an Image determines its height (in pixels) on the screen when displayed.	Instance Attribute
imageURL	image Location	--	This attribute of Image stores the location of the Image in question, <i>i.e.</i> , its URL (or URLs, if the Document is replicated in multiple locations).	Instance Attribute
imageWidth	horizontal Precision	--	The imageWidth attribute of an Image determines its width (in pixels) on the screen when displayed.	Instance Attribute
isEAGLESCompliant	--	--	The isEAGLESCompliant attribute of a Linguistic Tagset states whether this Linguistic Tagset was designed following the corresponding EAGLES recommendations or not.	Instance Attribute
isISOCompliant	--	--	This attribute of Tagset makes explicit whether it was designed following a suitable ISO standard or not.	Instance Attribute
isMarkupLanguage	isAnnotation Language	--	This attribute of Formal Language details whether the purpose of the language is to include annotations in a Document or not.	Instance Attribute
isStandardCompliant	--	--	This attribute of Tagset states whether the Tagset in question was designed following a suitable standard or not.	Instance Attribute
labelCode	--	--	This attribute details the (short) sequence of characters that constitute the Label (Annotation) under consideration in a Marked-Up Document for a particular Formal (mark-up) Language.	Instance Attribute
modificationDate	--	--	The modificationDate attribute of a Document details when it was (last) modified.	Instance Attribute
parameterName	--	--	When an Applet needs any kind of variable input values (the parameters or arguments of the Applet) for its execution, an identifier (name) for each parameter must be provided. This (multi-valued) attribute stores the names of the different parameters of the corresponding Applet.	Instance Attribute
parameterValue	--	--	This (multi-valued) attribute complements the information stored in the parameterName attribute of an Applet. In particular, it stores the values that each of the parameters included in the parameterName attribute of the Applet must take when it is run.	Instance Attribute
tripleID	--	--	In order to have perfectly fixed the particular Unit, Attribute and Value that constitute a particular Unit-Attribute-Value Triple, they are linked together by means of their corresponding identifiers, namely unitID, attributeID, valueID and	Instance Attribute

Name	Synonyms	Acronyms	Description	Type
			tripleID.	
typeOfLabel	--	--	This attribute of Annotation (Label) makes explicit the different purposes and/or functions for which the Label is included to mark-up a certain text Unit; for example, its format (italics, bold, etc.), its layout (table, cell, etc.) or some linguistic information (NP, PP, etc.).	Instance Attribute
unitID	--	--	In order to have perfectly fixed the particular Unit, Attribute and Value that constitute a particular Unit-Attribute-Value Triple, they are linked together by means of their corresponding identifiers, namely unitID, attributeID, valueID and tripleID.	Instance Attribute
valueID	--	--	In order to have perfectly fixed the particular Unit, Attribute and Value that constitute a particular Unit-Attribute-Value Triple, they are linked together by means of their corresponding identifiers, namely unitID, attributeID, valueID and tripleID.	Instance Attribute

Table 12: OIO glossary of terms (extracted from WebODE) – *ad hoc* relations

Name	Synonyms	Acronyms	Description	Type
Ascribes (<i>Linguistic Annotation Layer, Linguistic Attribute</i>)	--	--	--	Relation
Assigns (<i>Linguistic Annotation Layer, Linguistic Value</i>)	--	--	--	Relation
BelongsTo (<i>Linguistic Annotation, Linguistic Annotation Layer</i>)	--	--	--	Relation
DealsWith (<i>Linguistic Annotation Layer, Linguistic Unit</i>)	--	--	--	Relation
Describes (<i>Annotation, Content</i>)	--	--	--	Relation
Expresses (<i>Linguistic Unit, Unit</i>)	--	--	--	Relation
Has(<i>Unit, Attribute</i>)	--	--	--	Relation
HasInputLanguage (<i>Annotation Tool, Natural Language</i>)	--	--	--	Relation
HasOutputLanguage (<i>Annotation Tool, Formal Language</i>)	--	--	--	Relation
IsAttachedTo (<i>Annotation, Unit</i>)	--	--	--	Relation
IsWrittenIn (<i>Document, Language</i>)	--	--	--	Relation
Takes (<i>Attribute, Value</i>)	--	--	--	Relation
MarkedUpWith (<i>Marked-Up Document, Formal Language</i>)	--	--	--	Relation
Performs (<i>Annotator, Annotation</i>)	--	--	--	Relation
UsesOntology (<i>Annotation Tool, Ontology</i>)	--	--	--	Relation
UsesTagset (<i>Annotator, Tagset</i>)	--	--	--	Relation

Table 13: Instances of the OIO Linguistic Annotation Tool Instance Set (extracted from WebODE)

Name	Synonyms	Acronyms	Description
Catalan	--	--	--
DataLexica Morphological Analyser	--	--	A Linguistic Annotation Tool developed by Bitext (www.bitext.com) that provides Lemma-related and morphological information about its input. More concretely, it is a category non-disambiguating tool, which gives all the possible analyses of a Word as output. The input languages that this tool can recognise by now are Spanish, English and Catalan; the releases for French, German and Italian are under development.
DataLexica Tagset	--	--	--
Dutch	--	--	--

Name	Synonyms	Acronyms	Description
English	--	--	--
FDG Parser	--	--	A Linguistic Annotation Tool developed by Connexor (www.connexor.com) that not only gives Lemma-related and morphosyntactic information about its input but also the functional dependencies that hold between the words in the input sentences. The possible input languages for this tool are English, French, German, Spanish, Italian, Dutch, Finnish and Swedish.
FDG Tagset	--	--	--
Finnish	--	--	--
French	--	--	--
German	--	--	--
Italian	--	--	--
OntoTag Tagset	--	--	--
OntoTagger System	--	--	A Hybrid (linguistic and ontological) Annotation Platform developed within the OEG (Ontology Engineering Group) of the UPM (Universidad Politécnica de Madrid). It is intended to merge annotations at least at the morpho-syntactic and the semantic levels and for Spanish with reference to OntoTag's ontologies as well as other possible ontologies and resources, such as EuroWordNet.
OWL	--	--	--
RDF	--	--	--
Spanish	--	--	--
Swedish	--	--	--
UMurcia POS Tagger	--	--	A Linguistic Annotation Tool developed by the LACELL (Lingüística Aplicada Computacional, Enseñanza de Lenguas y Lexicografía) research group of the Murcia University (whose leader is Dr. Aquilino Sánchez Pérez). It analyses and tags words with a high precision with both Lemma-related and morphosyntactic information for Spanish, giving also some slight semantic information for adverbs.
UMurcia Tagset	--	--	--
XML	--	--	--

4.1.2.2 OIO CONCEPT TAXONOMIES

Once the glossary of terms has been built (that is, once the main terms to be included in the ontology have been identified and classified), there comes the time to arrange the concepts of this glossary into one or more **concept taxonomies**⁷³ and/or **meronomies**⁷⁴.

As far as taxonomies are concerned, as commented above, they are built out of the *Subclass-Of*, *Disjoint-Decomposition*, *Exhaustive-Decomposition* and *Partition* relations. Thus, (i) the different instances of the *Subclass-Of* relation that hold in the OIO are shown in Table 14; (ii) its one and only

⁷³ **Taxonomies** are built out of the well-known IsA relation, which associates an entity (called the *hyperonym* concept) of a certain type to another entity (called the *hyponym* concept) of a more general type. Taxonomies, thus, capture the type/subtype relations of a domain (EAGLES, 1999).

⁷⁴ **Meronomies** describe the Part-Whole (or *Part-Of*) relations that hold between the concepts of the domain being modelled (EAGLES, 1999).

disjoint-decomposition is presented in Table 15; and (iii) the different *exhaustive-decompositions* formalised in this ontology have been included in Table 16.

As far as meronomies are concerned, they are formalised in METHONTOLOGY (and, thus, in WebODE) by means of *Part-Of* relations (which can be either transitive or intransitive). The different instances of the (*Transitive*) *Part-Of* relation that have been found to hold between the concepts shown in the OIO glossary of terms are shown in Table 17 (no *Intransitive Part-Of* relation has been formalised within this ontology).

Table 14: OIO Subclass-Of relations (extracted from WebODE)

Source (Specialization)	Target (Generalization)
Hybrid Annotation	Linguistic Annotation
OntoTag-Annotated Document	Web Document
Hybrid Annotation	Ontological Annotation
Unit-Attribute-Value Triple	Subject-Predicate-Object Triple
Hybrid Annotation	Unit-Attribute-Value Triple
Hybridly Annotated Document	Ontologically Annotated Document
Hybridly Annotated Document	Linguistically Annotated Document
OntoTag-Annotated Document	Hybridly Annotated Document
Hybrid Tagset	Linguistic Tagset
Ontological Term Set	Ontological Tagset
Hybrid Tagset	Ontological Term Set
Hybrid Annotation Tool	Ontological Annotation Tool
Hybrid Annotation Tool	Linguistic Annotation Tool

Table 15: OIO Disjoint-Decompositions (extracted from WebODE)

Group	Group components	Target
Language Group	Formal Language	Language
	Natural Language	

Table 16: OIO Exhaustive-Decompositions (extracted from WebODE)

Group	Group components	Target
Value Group	Linguistic Value	Value
	Other Value	
Unit Group	Linguistic Unit	Unit
	Other Unit	
Tagset Group	Linguistic Tagset	Tagset
	Ontological Tagset	
	Other Tagset	
Marked-Up Document Group	Linguistically Annotated Document	Marked-Up Document
	Ontologically Annotated Document	
	Other Marked-Up Document	
Document Group	Marked-Up Document	Document
	Non-Marked-Up Document	
	Web Document	
Content Group	Image	Content
	Applet	
	Other Content	
	Plain Text	

Group	Group components	Target
Attribute Group	Linguistic Attribute	Attribute
	Other Attribute	
Annotator Group	Annotation Tool	Annotator
	Human Annotator	
Annotation Tool Group	Linguistic Annotation Tool	Annotation Tool
	Ontological Annotation Tool	
Annotation Group	Linguistic Annotation	Annotation
	Ontological Annotation	
	Other Annotation	
	Subject-Predicate-Object Triple	

Table 17: OIO *Part-Of* relations (extracted from WebODE)

Source (Meronym)	Target (Holonym)
Ontological Term	Ontological Term Set
Annotation	Marked-Up Document
Linguistic Annotation Layer	Linguistic Level
Ontological Term Set	Ontology
Unit	Unit-Attribute-Value Triple
Attribute	Unit-Attribute-Value Triple
Value	Unit-Attribute-Value Triple
Document	Corpus
Hybrid Annotation	Hybrid Tagset
Ontological Annotation	Ontological Tagset
Linguistic Annotation	Linguistic Tagset
Annotation	Tagset
Content	Document
Ontological Term	Ontological Annotation

4.1.2.3 OIO *Ad Hoc* RELATIONSHIPS

The next step, once the concepts of the ontology have been arranged into taxonomies and/or meronomies, consists of two consecutive (sub-)steps: (i) finding out the different *ad hoc* (that is, non-taxonomic and non-meronymic) relationships that hold between them and (ii) building their corresponding *ad hoc* binary relation diagrams.

The resulting OIO *ad hoc* binary relation diagrams are shown in Figure 8. A suitable description for the relationships shown in these figures can be found in the glossary of terms (Table 9 – Subsection 4.1.2.1).

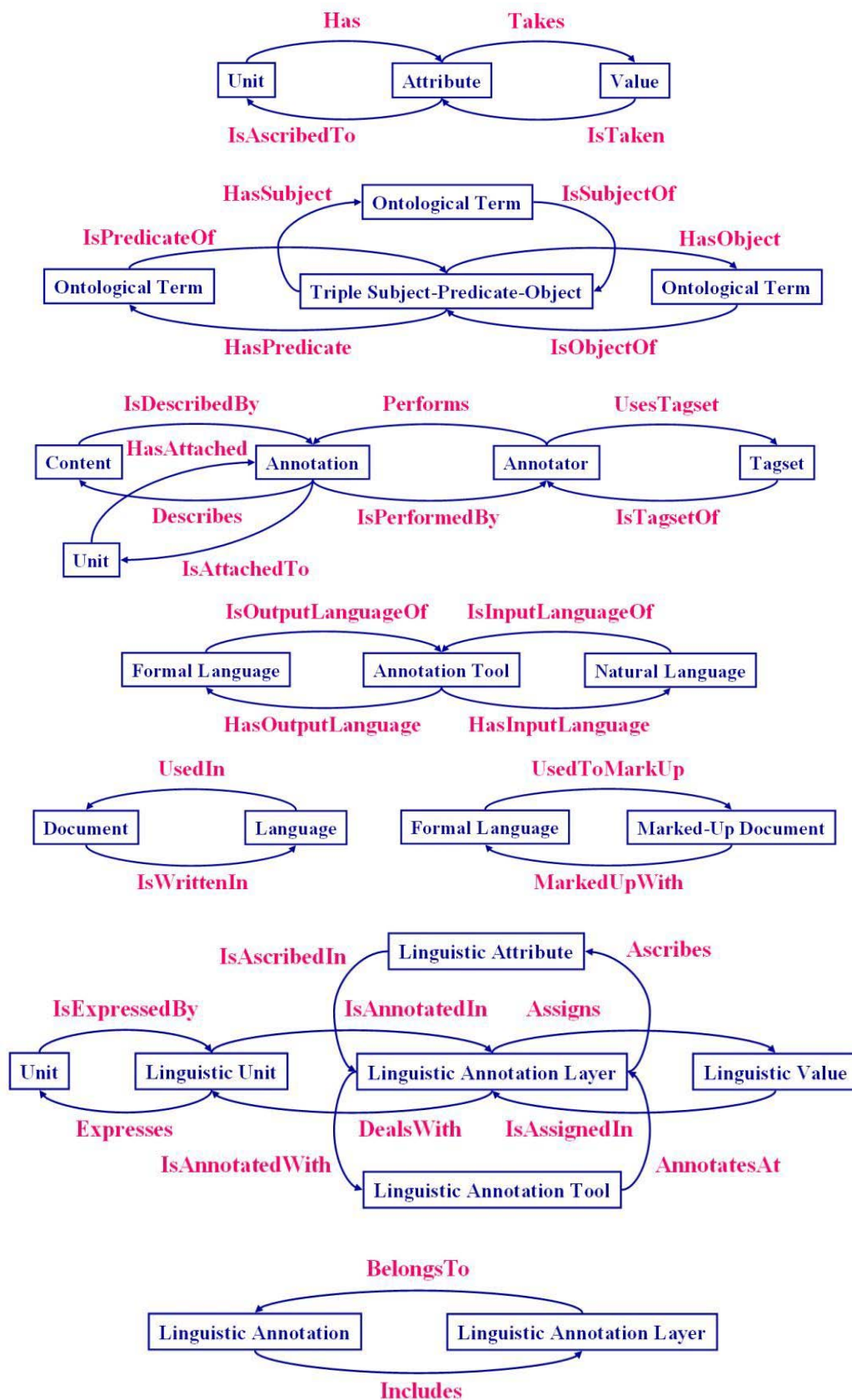


Figure 8: OIO's *ad hoc* relationships

4.1.2.4 OIO CONCEPT DICTIONARY

After detailing the *ad hoc* relations that hold between the concepts in the domain, the **concept dictionary** of the ontology is built. The concept dictionary summarises, for each concept, (i) the information about its relevant instances, (ii) its instance and class attributes and (iii) the *ad hoc* relations whose domain is the concept in question. The OIO concept dictionary (extracted from WebODE) is shown in Table 18 (page 92)⁷⁵.

Table 18: OIO concept dictionary (extracted from WebODE)

Concept name	Synonyms	Instances	Class attributes	Instance attributes	Relations
Annotation	Mark-Up Tag Label Metadata	--	--	annotationID annotatorID contentID labelCode typeOfLabel	Describes IsAttachedTo
Annotation Tool	Mark-Up Tool Tagger	--	--	--	HasInputLanguage HasOutputLanguage
Annotator	--	--	isHuman	annotatorID	Performs UsesTagset
Applet	--	--	--	alternativeText appletCode appletCodeBase appletHeight appletName appletWidth parameterName parameterValue	--
Attribute	Feature Property Quality Characteristic	--	--	attributeID	Takes
Content	--	--	--	contentID	--
Corpus	Text Corpus	--	--	--	--
Document ⁷⁶	File	--	--	Author creationDate documentURL modificationDate	IsWrittenIn
Formal Language	Artificial Language	OWL RDF XML	isNatural	isMarkUpLanguage	--
Human Annotator	--	--	--	--	--
Hybrid Annotation	--	--	conceptType	--	--
Hybrid Annotation Tool	--	OntoTagger System	conceptType	--	--
Hybrid Tagset	--	OntoTag Tagset	conceptType	--	--
Hybridly Annotated Document	Hybridly Marked- Up Document Hybridly Tagged Document	--	--	--	--

⁷⁵ The column of term acronyms has been removed for the sake of space.

⁷⁶ Several other attributes have already been identified for this concept in the literature. The main (and standard) ones are included in the Dublin Core metadata set (http://en.wikipedia.org/wiki/Dublin_Core) and in the Corpus Encoding Standard (CES – <http://www.cs.vassar.edu/CES/>; <http://www.cs.vassar.edu/CES/CES1-3.html>) and the Corpus Encoding Standard for XML (XCES – <http://www.xces.org/>). However, only those that appeared in the ODECORPUS-Entertainment corpus have already been included in the OIO. However, it is intended to extend the OIO with the attributes included in these standards as soon as possible.

Concept name	Synonyms	Instances	Class attributes	Instance attributes	Relations
Image	Picture Figure Pic	--	--	imageFormat imageHeight imageURL imageWidth	--
Language	--	--	--	--	--
Linguistic Annotation	--	--	conceptType	--	BelongsTo
Linguistic Annotation Layer	--	--	--	--	Ascribes Assigns DealsWith
Linguistic Annotation Tool	--	DataLexica Morphological Analyser FDG Parser UMurcia POS Tagger	conceptType	--	--
Linguistic Attribute	--	--	conceptType	--	--
Linguistic Level	--	--	--	--	--
Linguistic Tagset	--	DataLexica Tagset FDG Tagset UMurcia Tagset	conceptType	isBNCALike isCLAWSALike isEAGLESCompliant isSUSANNEALike	--
Linguistic Unit	--	--	conceptType	--	Expresses
Linguistic Value	--	--	conceptType	--	--
Linguistically Annotated Document	Linguistically Marked-Up Document Linguistically Tagged Document	--	--	--	--
Marked-Up Document	Annotated Document Tagged Document	--	--	--	MarkedUpWith
Natural Language	--	Catalan Dutch English Finnish French German Italian Spanish Swedish	isNatural	--	--
Non-Marked-Up Document	Non-Annotated Document Untagged Document	--	--	--	--
Ontological Annotation	Ontology-Based Annotation	--	conceptType	--	--
Ontological Annotation Tool	--	--	conceptType	--	--
Ontological Tagset	Ontology-Based Tagset	--	conceptType	--	--
Ontological Term	Ontological Component	--	--	--	--
Ontological Term Set	--	--	--	--	--
Ontologically Annotated Document	Ontologically Marked-Up Document Ontologically Tagged Document	--	--	--	--
Ontology	--	--	--	--	--

Concept name	Synonyms	Instances	Class attributes	Instance attributes	Relations
OntoTag- Annotated Document	OntoTag-Marked- Up Document OntoTag-Tagged Document	--	--	--	--
Other Annotation	--	--	--	--	--
Other Attribute	--	--	--	--	--
Other Content	--	--	--	--	--
Other Marked-Up Document	Other Annotated Document Other Tagged Document	--	--	--	--
Other Tagset	--	--	--	--	--
Other Unit	--	--	--	--	--
Other Value	--	--	--	--	--
Plain Text	--	--	--	backgroundColour fontColour fontSize fontType	--
Subject- Predicate-Object Triple	Triple Subject- Predicate-Object	--	--	--	--
Tagset	--	--	--	isISOCompliant isStandardCompliant	--
Unit	--	--	--	unitID	Has
Unit-Attribute- Value Triple	Triple Unit- Attribute-Value	--	--	attributeID tripleID unitID valueID	--
Value	--	--	--	valueID	--
Web Document	Web Page	--	--	--	--

4.1.2.5 OIO DETAILED TABLES

Once the concept dictionary has been built, an additional set of four tables must be generated as well. Each of these tables contains a detailed description of the terms included in the concept dictionary, namely (i) its *ad hoc* binary relations, (ii) its instance attributes, (iii) its class attributes, and (iv) its constants of the ontology. These detailing tables are presented in Subsections 4.1.2.5.1, 4.1.2.5.2, 0 and 4.1.2.5.4, respectively.

4.1.2.5.1 OIO *Ad Hoc* Binary Relation Table

An *ad hoc* binary relation table shows, for each *ad hoc* binary relation of the concept dictionary, (i) its name, (ii) the names of its source and target concepts, (iii) its (maximum) cardinality⁷⁷, (iv) its inverse relation and (v) its mathematical properties (for instance, reflexivity, symmetry and/or transitivity).

⁷⁷ The (maximum) cardinality of a relationship shows mainly whether a source concept can have no more than one corresponding target concept (cardinality = 1) or more than one (cardinality = N).

The detailed description of the OIO *ad hoc* binary relationships is included in Table 19.

Table 19: OIO *ad hoc* relationships (extracted from WebODE)

Relation name	Source concept	Source cardinality (Max)	Target concept	Mathematic properties	Inverse relation
Ascribes	Linguistic Annotation Layer	n	Linguistic Attribute	Antisymmetrical	To be implemented
Assigns	Linguistic Annotation Layer	n	Linguistic Value	Antisymmetrical	
BelongsTo	Linguistic Annotation	n	Linguistic Annotation Layer	Antisymmetrical	
DealsWith	Linguistic Annotation Layer	n	Linguistic Unit	Antisymmetrical	
Describes	Annotation	n	Content	Antisymmetrical	
Expresses	Linguistic Unit	n	Unit	Antisymmetrical	
Has	Unit	n	Attribute	Antisymmetrical	
HasInputLanguage	Annotation Tool	n	Natural Language	Antisymmetrical	
HasOutputLanguage	Annotation Tool	n	Formal Language	Antisymmetrical	
IsAttachedTo	Annotation	n	Unit	Antisymmetrical	
IsWrittenIn	Document	n	Language	Antisymmetrical	
MarkedUpWith	Marked-Up Document	n	Formal Language	Antisymmetrical	
Performs	Annotator	n	Annotation	Antisymmetrical	
Takes	Attribute	n	Value	Antisymmetrical	
UsesTagset	Annotator	n	Tagset	Antisymmetrical	

4.1.2.5.2 OIO Instance Attribute Table

Each instance attribute of the concept dictionary is described in detail in a row of the **instance attribute table** of the ontology. Hence, each row of this table contains the particular information about (i) the name of the attribute, (ii) the concept it belongs to, (iii) its value type, (iv) its measurement unit, precision and range of values (when its value type is numerical), (v) its default values (if any exists), (vi) its minimum and maximum cardinality⁷⁸, (vii) the instance attributes, the class attributes and the constants used to infer the values of the attribute; (viii) the attribute values that can be inferred using the values of this attribute; (ix) the formulae or rules that allow inferring the (some) values of the attribute, and (x) the references used to define the attribute.

The detailed description of the OIO instance attributes is shown in Table 20.

⁷⁸ The **minimum (maximum) cardinality of an (instance) attribute** is the minimum (maximum) number of values that attribute can have at the same time.

Table 20: OIO instance attribute table (extracted from WebODE)

Instance attribute name	Concept name	Value type	Measurement Unit	Precision	Value range	Default value	Cardinality
annotationID	Annotation	String	--	--	--	--	(1, 1)
annotatorID	Annotation	String	--	--	--	--	(1, 1)
contentID	Annotation	String	--	--	--	--	(1, 1)
labelCode	Annotation	String	--	--	--	--	(1, n)
typeOfLabel	Annotation	String	--	--	--	--	(1, 1)
annotatorID	Annotator	String	--	--	--	--	(1, 1)
alternativeText	Applet	String	--	--	--	--	(0, 1)
appletCode	Applet	String	--	--	--	--	(1, 1)
appletCodeBase	Applet	String	--	--	--	--	(0, n)
appletHeight	Applet	Cardinal	Pixel	1	--	480	(1, 1)
appletName	Applet	String	--	--	--	--	(1, 1)
appletWidth	Applet	Cardinal	Pixel	1	--	640	(1, 1)
parameterName	Applet	String	--	--	--	--	(0, n)
parameterValue	Applet	String	--	--	--	--	(0, n)
attributeID	Attribute	String	--	--	--	--	(1, 1)
contentID	Content	String	--	--	--	--	(1, 1)
author	Document	String	--	--	--	--	(0, n)
creationDate	Document	Date	--	--	--	--	(1, 1)
documentURL	Document	URL	--	--	--	--	(1, n)
modificationDate	Document	Date	--	--	--	--	(0, n)
isMarkUpLanguage	Formal Language	Boolean	--	--	--	--	(0, 1)
imageFormat	Image	String	--	--	--	--	(1, 1)
imageHeight	Image	Cardinal	Pixel	1	--	--	(1, 1)
imageURL	Image	URL	--	--	--	--	(1, n)
imageWidth	Image	Cardinal	Pixel	1	--	--	(1, 1)
isEAGLESCompliant	Linguistic Tagset	Boolean	--	--	--	--	(0, 1)
backgroundColour	Plain Text	String	--	--	--	#FFFFFF	(1, 1)
fontColour	Plain Text	String	--	--	--	#000000	(1, 1)
fontSize	Plain Text	Cardinal	Pixel	--	1 .. 100	12	(1, 1)
fontType	Plain Text	String	--	--	--	Times New Roman	(1, 1)
isISOCompliant	Tagset	Boolean	--	--	--	--	(0, 1)
isStandardCompliant	Tagset	Boolean	--	--	--	--	(0, 1)
unitID	Unit	String	--	--	--	--	(1, 1)
attributeID	Unit-Attribute-Value Triple	String	--	--	--	--	(1, 1)
tripleID	Unit-Attribute-Value Triple	String	--	--	--	--	(1, 1)
unitID	Unit-Attribute-Value Triple	String	--	--	--	--	(1, 1)
valueID	Unit-Attribute-Value Triple	String	--	--	--	--	(1, 1)
valueID	Value	String	--	--	--	--	(1, 1)

4.1.2.5.3 OIO Class Attribute Table

METHONTOLOGY (Gómez-Pérez *et al.*, 2004) recommends providing each class attribute of the concept dictionary with its detailed description in a row of the **class attribute table** of the ontology. Each of these rows contains, like the ones for instance attributes, the following information: (i) the name of the class attribute it refers to, (ii) the name of the concept to which the attribute is ascribed, (iii) its value type, (iv) its actual value, measurement unit and precision (when its value type is numerical), (v) its cardinality, (vi) the instance attributes whose values can be inferred with the value of this class attribute, etc.

The detailed description of the OIO class attributes is shown in Table 21.

Table 21: OIO class attribute table (extracted from WebODE)

Attribute name	Values	Concept name	Value type	Measurement unit	Precision	Cardinality
isHuman	false	Annotator	String	--	--	(0,1)
isNatural	false	Formal Language	Boolean	--	--	(0,1)
isHuman	true	Human Annotator	String	--	--	(0,1)
conceptType	LINGUISTIC ONTOLOGICAL	Hybrid Annotation	String	--	--	(0,n)
conceptType	LINGUISTIC ONTOLOGICAL	Hybrid Annotation Tool	String	--	--	(0,n)
conceptType	LINGUISTIC ONTOLOGICAL	Hybrid Tagset	String	--	--	(0,n)
conceptType	LINGUISTIC	Linguistic Annotation	String	--	--	(0,n)
conceptType	LINGUISTIC	Linguistic Annotation Tool	String	--	--	(0,n)
conceptType	LINGUISTIC	Linguistic Attribute	String	--	--	(0,n)
conceptType	LINGUISTIC	Linguistic Tagset	String	--	--	(0,n)
conceptType	LINGUISTIC	Linguistic Unit	String	--	--	(0,n)
conceptType	LINGUISTIC	Linguistic Value	String	--	--	(0,n)
isNatural	true	Natural Language	Boolean	--	--	(0,1)
conceptType	ONTOLOGICAL	Ontological Annotation	String	--	--	(0,n)
conceptType	ONTOLOGICAL	Ontological Annotation Tool	String	--	--	(0,n)
conceptType	ONTOLOGICAL	Ontological Tagset	String	--	--	(0,n)

4.1.2.5.4 OIO Constant Table

No constants were included in the concept dictionary of the OIO. Thus, this task was not performed for this ontology either and, therefore, it also lacks a constant table. However, had any constant been included in the OIO, its detailed description –consisting of (i) its name, (ii) its value type (a number, a string, etc.), (iii) its value (and measurement unit for numerical constants) and (iv) the attributes that can be inferred using this constant– should have been included in the **constant table** of the ontology.

4.1.2.6 OIO FORMAL AXIOMS AND RULES

After including all the concepts in the ontology and describing them in detail, together with their associated taxonomic (and/or meronymic) relations, attributes, *ad hoc* relations and constants, their corresponding formal axioms and rules must be defined as well. Formal axioms and rules are used, *e.g.*, for constraint checking and for inferring the values of some attributes. Hence, they are very significant components in heavyweight ontologies.

4.1.2.6.1 OIO Rule Table

Rules are generally used to infer knowledge in the ontology, such as attribute values, relation instances, etc. For each rule in the ontology, the following information should be included in its **rule table**: (i) the name of the rule, (ii) its natural language description, (iii) the expression that describes formally the rule, (iv) the concepts, attributes and (ad hoc) relations to which the rule refers, and (v) the variables used in the formal expression of the rule.

This task was not performed for the (OIO), since no rule was identified during its development. Accordingly, the OIO lacks a rule table.

4.1.2.6.2 OIO Formal Axiom Table

As defined above, formal axioms are logical expressions that are always true and are normally used to specify constraints in the ontology. METHONTOLOGY proposes to generate a **formal axiom table** that describes them in detail. This table provides, for each formal axiom of the ontology, (i) its name, (ii) its natural language description, (iii) the logical expression that describes formally the axiom (expressed in first order logic), (iv) the concepts, attributes and (*ad hoc*) relations to which the axiom refers, and (v) the variables used in its logical expression.

OIO's formal axioms are distributed between Table 22 and Table 23.

Table 22: OIO formal axiom table (1 of 2)

Axiom Name	Description	Expression	Concepts	Referred Attributes	Ad hoc binary relations	Variables
OIO.01	If there exists a Unit-Attribute-Value Triple, there must exist a Unit, an Attribute and a Value to constitute it.	$\text{exists}(\text{?T}, \text{?N}, \text{?I}, \text{?L}) \text{ ([Unit-Attribute-Value Triple]}(\text{?T}) \text{ and } [\text{unitID]}(\text{?T}, \text{?N}) \text{ and } [\text{attributeID]}(\text{?T}, \text{?I}) \text{ and } [\text{valueID]}(\text{?T}, \text{?L})) \rightarrow \text{exists}(\text{?U}, \text{?A}, \text{?V}) \text{ ([Linguistic Unit]}(\text{?U}) \text{ and } [\text{unitID]}(\text{?U}, \text{?N}) \text{ and } [\text{Transitive Part-Of}] (\text{?U}, \text{?T}) \text{ and } [\text{Linguistic Attribute}] (\text{?A}) \text{ and } [\text{attributeID}] (\text{?U}, \text{?I}) \text{ and } [\text{Transitive Part-Of}] (\text{?A}, \text{?T}) \text{ and } [\text{Linguistic Value}] (\text{?V}) \text{ and } [\text{valueID}] (\text{?U}, \text{?L}) \text{ and } [\text{Transitive Part-Of}] (\text{?V}, \text{?T}))$	Unit-Attribute-Value Triple Linguistic Unit Linguistic Attribute Linguistic Value	unitID attributeID valueID	--	?T ?N ?I ?L ?U ?A ?V
OIO.02	If there exists an Annotation, there must exist an Annotator that Performs it, and a Content, which the Annotation Describes.	$\text{forall}(\text{?A}, \text{?O}, \text{?T}) \text{ ([Annotation]}(\text{?A}) \text{ and } [\text{annotatorID}] (\text{?O}) \text{ and } [\text{contentID}] (\text{?T}) \rightarrow \text{exists}(\text{?N}, \text{?C}) \text{ ([Annotator]}(\text{?N}) \text{ and } [\text{annotatorID}] (\text{?N}, \text{?O}) \text{ and } [\text{Performs}] (\text{?N}, \text{?A}) \text{ and } [\text{Content}] (\text{?C}) \text{ and } [\text{contentID}] (\text{?C}, \text{?T}) \text{ and } [\text{Describes}] (\text{?A}, \text{?C}))$	Annotation Annotator Content	annotatorID contentID	Performs Describes	?A ?O ?T ?N ?C
OIO.03	A Linguistic Annotation Tool Uses a Linguistic Tagset (not any -other- type of Tagset).	$\text{forall}(\text{?L}, \text{?T}) \text{ ([Linguistic Annotation Tool]}(\text{?L}) \text{ and } [\text{Tagset}] (\text{?T}) \text{ and } [\text{UsesTagset}] (\text{?L}, \text{?T}) \rightarrow [\text{Linguistic Tagset}] (\text{?L}))$	Linguistic Annotation Tool Tagset Linguistic Tagset	--	UsesTagset	?L ?T

Table 23: OIO formal axiom table (2 of 2)

Axiom Name	Description	Expression	Concepts	Referred Attributes	binary relations	Variables
OIO.04	An Ontological Annotation Tool <i>Uses</i> an Ontological Tagset (not any -other- type of Tagset).	forall(?O, ?T) (([Ontological Annotation Tool](?O) and [Tagset](?T) and [UsesTagset](?O, ?T)) → [Ontological Tagset](?O))	Ontological Annotation Tool Tagset Ontological Tagset	--	UsesTagset	?O ?T
OIO.05	A Hybrid Annotation Tool <i>Uses</i> a Hybrid Tagset (not any -other- type of Tagset).	forall(?H, ?T) (([Hybrid Annotation Tool](?H) and [Tagset](?T) and [UsesTagset](?H, ?T)) → [Hybrid Tagset](?H))	Hybrid Annotation Tool Tagset Hybrid Tagset	--	UsesTagset	?H ?T
OIO.06	A Linguistic Unit <i>Has</i> Linguistic Attributes (not any -other- type of Attributes).	forall(?U, ?A) (([Linguistic Unit](?U) and [Attribute](?A) and [Has](?U, ?A)) → [Linguistic Attribute](?A))	Linguistic Unit Attribute Linguistic Attribute	--	Has	?U ?A
OIO.07	A Linguistic Attribute <i>Takes</i> Linguistic Values (not any -other- type of Values).	forall(?A, ?V) (([Linguistic Attribute](?A) and [Value](?V) and [Takes](?A, ?V)) → [Linguistic Value](?V))	Linguistic Attribute Value Linguistic Value	--	Takes	?A ?V

4.1.2.7 OIO INSTANCE TABLE

Once the conceptual model of the ontology has been created, the relevant instances that were included in the concept dictionary must be defined inside an **instance table**. For each instance, the following information should be provided in the table: (i) its name, (ii) the name of the concept it belongs to, (iii) its attributes (together with their corresponding values, if known), and (iv) its instanced relations (together with the instances to which it is related, when known).

The OIO instance table is shown in Table 24.

Table 24: OIO instance table

Instance Name	Concept name	Attribute / Instanced relation	Value(s) / Target Instance(s)
Catalan	Natural Language	--	--
DataLexica Morphological Analyser	Annotator	Performs	--
		UsesTagset	DataLexica Tagset
	Annotation Tool	HasInputLanguage	Catalan
			English
			French
			German
Italian			
Spanish			
DataLexica Tagset	Tagset	HasOutputLanguage	--
		isISOC compliant	false
	Linguistic Tagset	isStandardCompliant	false
		isEAGLESC compliant	false
Dutch	Natural Language	--	--
English	Natural Language	--	--
FDG Parser	Annotator	Performs	--
		UsesTagset	FDG Tagset
	Annotation Tool	HasInputLanguage	Dutch
			English
			Finnish
			French
			German
			Italian
Spanish			
Swedish			
FDG Tagset	Tagset	HasOutputLanguage	--
		isISOC compliant	false
	Linguistic Tagset	isStandardCompliant	false
		isEAGLESC compliant	false
Finnish	Natural Language	--	--
French	Natural Language	--	--
German	Natural Language	--	--
Italian	Natural Language	--	--
OntoTag Tagset	Tagset	isISOC compliant	true
		isStandardCompliant	true
	Linguistic Tagset	isEAGLESC compliant	true
	Hybrid Tagset	--	--
OntoTagger System	Annotator	Performs	--

Instance Name	Concept name	Attribute / Instanced relation	Value(s) / Target Instance(s)
		UsesTagset	OntoTag Tagset
		HasInputLanguage	Spanish
	Annotation Tool	HasOutputLanguage	OWL
			RDF
			XML
OWL	Formal Language	isMarkupLanguage	true
RDF	Formal Language	isMarkupLanguage	true
Spanish	Natural Language	--	--
Swedish	Natural Language	--	--
	Annotator	Performs	--
UMurcia POS Tagger		UsesTagset	UMurcia Tagset
	Annotation Tool	HasInputLanguage	Spanish
		HasOutputLanguage	--
	Tagset	isISOCompliant	false
UMurcia Tagset		isStandardCompliant	false
	Linguistic Tagset	isEAGLESCompliant	false
XML	Formal Language	isMarkupLanguage	true

4.1.2.8 THE OIO STATISTICS

To conclude this section, the OntoTag Integration Ontology statistics have been summarised in Table 25.

Table 25: The OIO statistics

OIO	CONCEPTS	INSTANCES	ATTRIBUTES	RELATIONS			RULES	AXIOMS	TOTAL
				TAX.	MERON.	AD HOC			
TOP-LEVEL	50	20	57	24	14	15	0	7	187
(MORPHO) SYNTACTIC LEVEL	0	0	0	0	0	0	0	0	0
SEMANTIC LEVEL	0	0	0	0	0	0	0	0	0
TOTAL	50	20	57	24	14	15	0	7	187
				53					

Thus far, the OntoTag Integration Ontology (OIO) has been described. In particular, the last eight subsections describe the components that it includes, that is, the top-level concepts that are subclassified and characterised in the remaining ontologies of OntoTag, as well as the most important (*ad hoc*) relations holding between them. To start with, the subclasses of the Linguistic Unit concept in the LUO, its corresponding attributes, *ad hoc* relations, rules and axioms are presented in the following section.

4.1.3. THE LINGUISTIC UNIT ONTOLOGY (LUO)

The Linguistic Unit Ontology (LUO) includes all the units (categories) identified for the different linguistic levels considered in OntoTag, that is, the syntactic, the morphosyntactic (subsumed in the previous one) and, to some extent, also the semantic level. This ontology is described in six subsections. The first one (Subsection 4.1.3.1) describes the top-level concepts of the LUO and their corresponding attributes. Each one of the following four subsections is devoted to a particular ontology module (Doran *et al.*, 2007) of this ontology. Hence, Subsection 4.1.3.2 describes the syntactic units, included in the syntactic module of the LUO (morphosyntactic units *inclusive*); and Subsection 4.1.3.3 presents the semantic units included in its semantic module. For the sake of clarity and space, each of these two subsections has been subdivided for its description into four subsections: (i) *concepts* and *taxonomy*, (ii) *attributes*, (iii) *ad hoc relations* and (iv) *rules* and *axioms*. Finally, Subsection 4.1.3.4 presents some global statistics about the elements included in this ontology.

4.1.3.1 THE CONCEPT LINGUISTIC UNIT, ITS SUBCLASSES AND ITS ATTRIBUTES

This subsection presents the top-level concept of the LUO, `Linguistic Unit`, and its main-subclasses, namely `Morphological Unit`, `Syntactic Unit`, `Semantic Unit`, `Discourse Unit` and `Pragmatic Unit`. They have been included in Table 26.

Concerning the `Morphological Unit`, `Discourse Unit` and `Pragmatic Unit` concepts, they have been added to the LUO for completeness sake and, thus, they have not been further subclassified and/or characterised in OntoTag's ontologies⁷⁹. However, the first and the second ones have at least one subclass that is a key concept within the present work, namely `Morphosyntactic Unit` (presented in Subsection 4.1.3.2.1) and `Named Entity` (presented in Subsection 4.1.3.3.1), respectively. Each of these subclasses stand on the interface between two levels: the morphological-syntactic in the first case, and the semantic and the discourse-related, in the second case.

Therefore, morphosyntactic units are morphological units and syntactic units at a time, as well as named entities are semantic units and discourse units. On the one hand, morphosyntactic units (*i.e.*, words, basically) are the highest rank type of morphological units. In addition, morphosyntactic units have a clear projection on the `Syntactic Level` and share some attributes, values and properties with other 'pure' syntactic units, as shown below. Therefore, they are also a type of `Syntactic`

⁷⁹ This falls out of the scope of the present work and has left for further research.

Unit. This entails that morphosyntactic units belong to both the `Morphological` and the `Syntactic Level`⁸⁰, which prevents these levels from being treated independently.

On the other hand, named entities are units with a clear semantics, which is neatly different from the semantics carried out by other types of morphosyntactic and syntactic units. Thus, they are clear semantic units. Nevertheless, from a discourse analysis point of view, they are also they very important units, since they are co-refered pervasively in texts, after being introduced, by means of cohesive devices, such as anaphorae and cataphorae. Accordingly, they can also be considered a particular type of discourse units. This entails that named entities belong to both the `Semantic` and the `Discourse Level`, which prevents these levels from being treated independently too.

This is a most relevant result as for this work, since **it refutes one of the hypotheses underlying the development of OntoTag, namely that language phenomena can be divided into levels that can be treated separately**. As for the present release of OntoTag’s ontologies, morphosyntactic units have been included in the syntactic module of the LUO. Yet, also the *Subclass-Of* relation that links `Morphosyntactic Unit` to `Morphological Unit` has been formalised within the LUO, as explained above. The same applies for `Named Entity`, which is a subclass of `Semantic Unit` and of `Discourse Unit`.

Table 26: The main concepts in the LUO and the taxonomical relations holding between them

Linguistic Unit	Morphological Unit
	Syntactic Unit
	Semantic Unit
	Discourse Unit
	Pragmatic Unit

The attributes associated to the concept `Linguistic Unit` are included in Table 27. This table shows their type (*i.e.*, whether it is an instance attribute or a class attribute) and the type of values that it can take. On the one hand, `Identifier` is used to designate univocally each `Linguistic Unit` found in an input document⁸¹. On the other hand, `Order Number` details the relative order of the `Linguistic Unit` with respect to the other units that constitute the higher-rank `Linguistic Unit` in which all of them are included.

⁸⁰ In fact, morphosyntactic units constitute the interface between the `Morphological` and the `Syntactic Level` (*cf.* <http://www.sil.org/linguistics/GlossaryOfLinguisticTerms/WhatIsAWord.htm>).

⁸¹ See the recommended mechanism for unit identifier construction in Subsection 5.4.3.2.

Table 27: (Ontological) Attributes associated to the concept `Linguistic Unit`

CONCEPT	ATTRIBUTE	ATTRIBUTE TYPE	VALUE TYPE
Linguistic Unit	Identifier	Instance attribute	STRING
	OrderNumber	Instance attribute	CARDINAL

The attributes associated to the different subclasses of `Linguistic Unit` are presented in the following subsections, together with the rules and the axioms added to the ontology to further specify the semantics of these concepts. Their *ad hoc* relations with other concepts (outside the LUO) have already been discussed in the description of the OIO (Subsection 4.1.2) and, hence, they do not need to be discussed here again.

After presenting the top-level concepts of the LUO, which formalise the most general types of linguistic units, it is the time to proceed with the presentation of its syntactic units (which include the morphosyntactic units). This type of units and their corresponding attributes, *ad hoc* relations, rules and axioms are introduced in the next subsection.

4.1.3.2 THE SYNTACTIC MODULE

The linguistic units belonging to both the morphosyntactic and the syntactic levels were formalised in `OntoTag` within this module of the LUO. It incorporates all the morphosyntactic and syntactic units identified in the EAGLES (1996a, 1996b) recommendations for morphosyntactic and syntactic annotation⁸². Yet, the linguistic knowledge included in these recommendations had to be re-structured beforehand, in order to determine what can be considered a `Linguistic Unit`, what a `Linguistic Attribute` and what a `Linguistic Value`.

In effect, the EAGLES (1996a) recommendations do not tell the linguistic terms dealing with unit subclassification from those dealing with inflection. Therefore, prior to their inclusion in `OntoTag`'s ontologies, these terms had to be divided into subclassifying attributes and inflective attributes. Then, each of these types of attributes was treated differently. On the one hand, subclassifying attributes gave rise to new concepts in the LUO; on the other hand, inflective attributes originated new concepts and/or instances in the LAO and in the LVO.

For example, the 'Type' attribute associated to nouns in EAGLES (1996a) is a subclassifying attribute, since its two values (that is, 'Common' and 'Proper') characterise the corresponding two subclasses of `Noun` (the concept in the LUO formalising nouns). Accordingly, these two values of the

⁸² The original term used to refer to units within EAGLES (1996a) is **categories**, which is also the preferred term in ISO/TC 37 standardisation subcommittees and, in particular, in the ISO/DCR (2008) international standard draft.

‘Type’ attribute originated the inclusion of two new concepts in the LUO, namely Common Noun and Proper Noun. More precisely, these two concepts constitute a *Partition* of Noun and they were formalised as such in the LUO. On the contrary, the ‘Singular’ and ‘Plural’ values of the EAGLES (1996a) ‘Number’ attribute (which can be ascribed to nouns, amongst other morphosyntactic units) do not subclassify nouns and, hence, ‘Number’ should be considered an inflective attribute. Accordingly, this EAGLES (1996a) attribute originated the inclusion of (i) a concept in the LAO (Number) and another in the LVO (Number Value) and (ii) two *Instances-Of* Number Value (Singular and Plural) in the latter ontology.

Thus, (1) each value of each subcategorising attribute in EAGLES (1996a) originated the inclusion of a new concept in the LUO (*i.e.*, a new Morphosyntactic Unit or Syntactic Unit); (2) each EAGLES (1996a) inflective attribute originated the inclusion of a new concept in the LAO and another one in the LVO; and (3) each EAGLES (1996a) value of each inflective attribute originated the inclusion of a new instance in the LVO.

Hence, the EAGLES (1996a; 1996b) subclassifying attributes and their corresponding values were formalised in the LUO, and not in the LAO and/or the LVO. Besides the ‘Type’ attribute mentioned above, which appears associated to several morphosyntactic categories, the EAGLES (1996a; 1996b) subclassifying attributes are (i) the ‘Function’ attribute associated to numerals and (ii) the ‘Auxiliary-function’ attribute associated to verbs. These attributes allow for a more precise subclassification of the LUO concepts, via *Subclass-Of*, *Exhaustive-Decomposition*, *Disjoint-Decomposition* and *Partition* relations, as with the ‘Type’ attribute of nouns mentioned above.

Consequently, the *concepts* representing the syntactic units of the LUO and the *taxonomical relations* holding between them are presented in the following subsection (4.1.3.2.1). The *attributes* characterising them are included in Subsection 4.1.3.2.2. The *ad hoc relations* holding between these syntactic elements are discussed in Subsection 4.1.3.2.3. Finally, the *axioms* and *rules* that constrain and limit their semantics are shown in Subsection 4.1.3.2.4.

4.1.3.2.1 The Syntactic Module Concepts and Taxonomy

The concepts of the LUO representing syntactic units (*i.e.*, the linguistic units that belong to the Morphosyntactic and the Syntactic Level), as well as (i) the taxonomical relations that hold between these concepts and (ii) the relations holding between them and the LUO top-level concepts are all summarised from Table 28 to Table 41.

Table 28 shows the main syntactic concepts in the LUO. The three major **syntactic units** (Phrase, Clause and Sentence) were already included and subcategorised in EAGLES (1996b).

Hence, they were incorporated as such in the LUO. To these three units, for the sake of completeness, another unit (Token) was added as well. **Token has two subcategories**, namely, Simple Token (or Morphosyntactic Unit) and MultiWord Token, depending on the number of words that constitute the unit. For example, ‘Inés Sastre’ is a Multiword Token, while ‘actriz’ (≡ ‘actress’) is a Simple Token. Whereas the existence of a Simple Token concept in the ontology requires no further explanation, the existence of Multiword Token needs a proper justification. The concept Multiword Token formalises the fact that, in certain languages, some morphosyntactic information, as well as some lexical elements (such as concepts and individuals, like ‘Harry Potter’, for example), need to be expressed by means of a multiword unit (a Multiword Token, in this case⁸³). This is why the concept Multiword Token was eventually added to the present ontology.

Table 28: The syntactic super-concepts in the LUO and their taxonomical relations

TOP-LEVEL CONCEPTS		SYNTACTIC CONCEPTS		
Linguistic Unit	Syntactic Unit	Token	Morphosyntactic Unit (Simple Token)	[Continued in Table 29]
			Multiword Token (MultiWord Token)	
			Phrase	[Continued in Table 40]
			Clause	
			Sentence	

The rest of the syntactic units presented in this subsection, distributed from Table 29 to Table 40, are subclasses of the classes presented in Table 28. Concerning Table 29, it contains the **subclasses of Morphosyntactic Unit**. They have been taken (after being re-structured) from EAGLES (1996a).

Table 29: The Morphosyntactic Unit subclasses and the taxonomical relations holding between them

Morphosyntactic Unit	Word	Adjective	[Continued in Table 30]
		Adposition	
		Adverb	[Continued in Table 31]
		Article	[Continued in Table 32]
		Conjunction	[Continued in Table 33]
		Interjection	
		Noun	[Continued in Table 34]
		Numeral	[Continued in Table 35]

⁸³ ‘Harry Potter’ is a Multiword Token at the Syntactic Level, but its projection on the Semantic Level is an instance of Named Entity (see Section 4.1.3.3.1).

		Pronoun/Determiner	[Continued in Table 36]
		Residual	[Continued in Table 37]
		Unique/Unassigned	
		Verb	[Continued in Table 38]
	Punctuation Mark	[Continued in Table 39]	
	Lemma		

As for Table 30, it presents the EAGLES-derived **subclasses of Adjective** and of **Adposition**. Concerning the subclassification of adpositions shown in this table, they are further subclassified in EAGLES (1996a) as circumpositions⁸⁴, postpositions⁸⁵, prepositions, and fused preposition-articles⁸⁶. Because of the different properties of these four categories, it was decided to rearrange them as shown in Table 30. Hence, the concept Adposition was subclassified into the concepts Proper Adposition and Fused Prep-At. The concept Proper Adposition, in turn, was subclassified into the three morphosyntactic categories Preposition, Postposition and Circumposition included in EAGLES (1996a).

Table 30: The Adjective and Adposition subclasses

Adjective	Cardinal Adjective ⁸⁷	Adposition	Proper Adposition
	Ordinal Adjective ⁸⁸		Fused Prep-At (Contraction)

Table 31 introduces the **subclasses of Adverb** derived from EAGLES (1996a), as well as a simplified representation of the taxonomical relations holding between them. A comment should be made about this subclassification. Both a Non Wh-Adverb and a Wh-Adverb can be either a General Adverb or a Pronominal Adverb. Therefore, this second-level subcategorisation of adverbs (*i.e.*, Non Wh-Adverb vs. Wh-Adverb) is rather independent with respect to their first-level subcategorisation (*i.e.*, General Adverb vs. Pronominal Adverb). Accordingly, it originates four different types of adverbs, namely general Non-Wh-adverbs⁸⁹, general Wh-adverbs⁹⁰, pronominal Wh-adverbs⁹¹ and pronominal Non-Wh-adverbs⁹². As shown in Table 31, only Wh-

⁸⁴ As, for example, the Circumposition ‘**um...herum**’ in the German Sentence ‘Reden wir nicht lange **um** die Sache **herum**’ (≡ ‘Let’s not talk much **about** that issue’).

⁸⁵ For instance, ‘**entlang**’ in the German Phrase ‘die Straße **entlang**’ (≡ ‘**along** the street’).

⁸⁶ For example, the Spanish word ‘**al**’ (= ‘a’ + ‘el’) and the German word ‘**zum**’ (= ‘zu’ + ‘dem’), which can be translated as ‘**to the**’ into English.

⁸⁷ Cardinal Adjective is a *Subclass-Of* both Adjective and Cardinal.

⁸⁸ Ordinal Adjective is a *Subclass-Of* both Adjective and Ordinal.

⁸⁹ Such as the English Adverb ‘**here**’.

⁹⁰ See in <http://www.ling.upenn.edu/histcorpora/annotation/pos-wh.htm> a description (with examples of the difference between general Wh-adverbs and pronominal Wh-adverbs).

⁹¹ See the previous footnote.

⁹² As, for instance, the German word ‘**dafür**’ in the Sentence ‘Wir haben kein Geld **dafür**’ (≡ ‘We have no money **for that**’).

adverbs (that is, general Wh-adverbs and pronominal Wh-adverbs), can be further subclassified as exclamatory, interrogative and relative adverbs.

Table 31: The Adverb subclasses and the taxonomical relations holding between them

Adverb	Degree Adverb			
	Particle Adverb			
	General Adverb	Non Wh-Adverb		
		Wh-Adverb	Exclamatory Adverb	
Interrogative Adverb				
Relative Adverb				
Pronominal Adverb ⁹³				

With respect to Table 32, it includes the EAGLES-derived **subclasses of Article**. Concerning Personal Article, it is a language-specific category that was not included in the EAGLES (1996a) recommendations. A Personal Article is a particular type of Article that precedes (person) proper nouns in Catalan. It can be realised (only in an informal register) by means of its Masculine Form, ‘en’, its Feminine Form, ‘na’, or its Contracted Form (n’). This unit was added to the LUO for the sake of completeness.

Table 32: The Article subclasses

Article	Definite Article
	Indefinite Article
	Partitive Article
	Personal Article ⁹⁴

As far as Table 33 is concerned, it shows the EAGLES-derived **subclasses of Conjunction**, as well as a simplified representation of the taxonomical relations holding between them. In this case, the original EAGLES (1996a) subclassification has been extended with the subcategorisation of the Simple Coordinating Conjunction concept, taken from Lázaro-Tusón (1990).

Table 33: The Conjunction subclasses and the taxonomical relations holding between them

Conjunction	Coordinating Conjunction	Correlative Coordinating Conjunction	
		Initial Coordinating Conjunction	
		Non-Initial Coordinating Conjunction	
	Simple Coordinating Conjunction	Adversative Conjunction	
		Copulative Conjunction	

⁹³ This concept is also a *Subclass-Of* Pronoun/Determiner.

⁹⁴ See <http://ca.wikipedia.org/wiki/Article>.

Subordinating Conjunction	Disjunctive Conjunction	
	Adverbial Subordinating Conjunction	
	Comparative Subordinating Conjunction	
	With Finite Subordinating Conjunction	
	With Infinite Subordinating Conjunction	
Substantive Subordinating Conjunction		

Concerning Table 34, it contains the **subclasses of Noun** included in EAGLES (1996a), as well as a simplified representation of the taxonomical relations holding between them.

Table 34: The Noun subclasses and the taxonomical relations holding between them

Noun	Common Noun	Countable Noun
		Mass Noun
	Proper Noun	

As for Table 35, it presents the EAGLES-derived **subclasses of Numeral**, as well as a simplified representation of the taxonomical relations holding between them.

Table 35: The Numeral subclasses and the taxonomical relations holding between them

Numeral	Cardinal	Cardinal Pronoun/Determiner	Cardinal Determiner
			Cardinal Pronoun
		Cardinal Adjective ⁹⁵	
	Ordinal	Ordinal Pronoun/Determiner	Ordinal Determiner
			Ordinal Pronoun
		Ordinal Adjective ⁹⁶	

Regarding Table 36, it introduces the EAGLES-derived **subclasses of Pronoun** and **Determiner**, as well as a simplified representation of the taxonomical relations holding between them.

Table 36: The Pronoun/Determiner subclasses and the taxonomical relations holding between them

Pronoun/ Determiner ⁹⁷	Demonstrative Pronoun/Determiner	Demonstrative Determiner	
		Demonstrative Pronoun	
	Exclamatory Pronoun		
	Indefinite	Indefinite Determiner	

⁹⁵ As commented above, Cardinal Adjective is a *Subclass-Of* both Adjective and Cardinal.

⁹⁶ As commented above, Ordinal Adjective is a *Subclass-Of* both Adjective and Ordinal.

⁹⁷ EAGLES (1996a) established a common major category for pronouns and determiners.

	Pronoun/Determiner	Indefinite Pronoun		
	Interrogative Pronoun/Determiner	Interrogative Determiner		
		Interrogative Pronoun		
	Numeral Pronoun/Determiner	Cardinal Pronoun/Determiner	Cardinal Determiner	
			Cardinal Pronoun	
		Ordinal Pronoun/Determiner	Ordinal Determiner	
			Ordinal Pronoun	
	Partitive Determiner			
	Personal Pronoun			
	Possessive Pronoun/Determiner	Possessive Determiner		
		Possessive Pronoun		
	Reciprocal Pronoun			
	Reflexive Pronoun			
Relative Pronoun/Determiner	Relative Determiner			
	Relative Pronoun			

With respect to Table 37, it includes the EAGLES-derived **subclasses of Residual**, as well as a simplified representation of the taxonomical relations holding between them.

Table 37: The Residual subclasses and the taxonomical relations holding between them

Residual	Abbreviation	Acronym
	Foreign Word	
	Formula	
	Symbol	
	Unclassified Residual	

As for Table 38, it presents the EAGLES-derived **subclasses of Verb**, as well as a simplified representation of the taxonomical relations holding between them.

Table 38: The Verb subclasses and the taxonomical relations holding between them

Verb	Auxiliary Verb	Modal Auxiliary Verb
		Primary Auxiliary Verb
	Main Verb	Copular Verb
		Predicative Verb
	Semi-Auxiliary Verb	Aspectual Semi-Auxiliary Verb
		Modal Semi-Auxiliary Verb

Concerning Table 39, it contains the EAGLES-derived **subclasses of Punctuation Mark** and their instances, as well as a simplified representation of the taxonomical relations holding between these subclasses. An extra hierarchical level was incorporated into this subclassification for those subclasses of Punctuation Mark not contemplated in EAGLES (1996a), in order to represent better these subcategories of Morphosyntactic Unit.

Table 39: The Punctuation Mark subclasses and instances

CONCEPTS		INSTANCES
Punctuation Mark ⁹⁸	Other Punctuation Mark	Period
		Comma
		Question Mark
		Exclamatory Mark
		Colon
		Semi-Colon
		Question Mark (Open)
		Exclamatory Mark (Open)
		Apostrophe
		Hyphen
		Dash
		Slash
		Backslash
		Simple Quotation Mark
	Simple Quotation Mark (Close)	
	Double Quotation Mark	Double Quotation Mark (Open)
		Double Quotation Mark (Close)
	Parenthesis	Parenthesis (Open)
		Parenthesis (Close)
	Bracket	Bracket (Open)
		Bracket (Close)

This concludes the formalisation of the morphosyntactic units derived from the EAGLES (1996a) recommendations within the LUO. The recommended attributes and values not used in the determination of these units do not subcategorise any of them. Therefore, they were considered proper syntactic attributes (*id.* values) in OntoTag and included in the LAO (*id.* the LVO). Yet, the ‘Use’ attribute, whose corresponding values are ‘Predicative’ and ‘Attributive’, and which was associated to ‘Adjective’ in EAGLES (1996a), constitutes an exception to this statement. This attribute was formalised in the LAO by means of the Use concept, after being conveniently extended and associated

98

not only with adjectives but also with other suitable units of the LUO. Besides, it was moved to the Semantic Level, where it takes its real meaning.

Finally, as far as Table 40 is concerned, it represents all the different types of Phrase, Clause and Sentence included in EAGLES (1996b). According to these recommendations, the class Phrase has been subclassified with respect to a rather traditional criterion, *i.e.*, the type of Morphosyntactic Unit that heads the Phrase. In this subclassification, there are four primary concepts, namely, Adjective Phrase, Adverb Phrase, Noun Phrase, and Verb Phrase. They represent the phrases that are headed, respectively, by adjectives, adverbs, nouns and verbs (that is, by the four types of open-class words). Hence, according to a purely lexical-semantic criterion, these four primary concepts would constitute an *Exhaustive-Decomposition* of Phrase on their own. However, following a straightforward syntactic approach, the concept Prepositional Phrase has been traditionally included in this subclassification as well, in order to represent those phrases headed by a Preposition. In addition, this subclassification has been traditionally considered a *Partition* (that is, both an *Exhaustive-Decomposition* and a *Disjoint-Decomposition* of the class in question) in the literature.

Table 40: The subclasses of Phrase, Clause and Sentence in the LUO

Phrase	Adjective Phrase
	Adverb Phrase
	Noun Phrase
	Verb Phrase
	Prepositional Phrase
Clause	Adverbial Clause ⁹⁹
	Comparative Clause
	Nominal Clause
	Relative Clause (Adjectival Clause)
Sentence	

As far as the taxonomical relations that hold between all these syntactic units are concerned, they have been formalised according to the same criteria applied in previous subsections. There are *Partitions*, *Disjoint-Decompositions*, *Exhaustive-Decompositions* and simple *Subclass-Of* relations. However, an exhaustive discussion of their determination is not included here for the sake of space.

⁹⁹ Further traditional subclassifications have been postulated for this Unit (such as Manner Adverbial Clause, Time Adverbial Clause, Location Adverbial Clause, Final Adverbial Clause, etc.). However, they are based on semantic grounds and, therefore, it seems more reasonable to characterise them by means of their corresponding annotations at the Semantic Level.

Additionally, the formalisation of all the relationships that hold for these syntactic units entailed the inclusion of the concept Lemma (Canonical Form)¹⁰⁰ in this module of the LUO.

Besides these taxonomical relations, there are some relevant *Part-Of* relations holding between these concepts as well. They have been included in Table 41. In this type of tables, by definition, the meronyms are considered to be *Part-Of* the holonyms.

Table 41: The *Part-Of* relations that hold for the syntactic concepts in the LUO

HOLONYM	MERONYM
Sentence	Clause
Clause	Phrase
Phrase	Token
Multiword Token	Morphosyntactic Unit (Simple Token)

Up to this point, the syntactic units contemplated in the LUO and their corresponding taxonomical relations have already been presented. There remains to present as well their attributes, *ad hoc* relations and their associated rules and axioms. This will be done in the next subsections.

4.1.3.2.2 The Syntactic Module Attributes

The attributes associated to the syntactic units of the LUO are presented in this subsection. All of them are *class attributes*. These attributes are included in Table 42, which shows, for each attribute, the unit to which it is ascribed and the values that it can take. Some of the attributes associated to the concepts in this table are associated to its subclasses as well. In these subclasses, these attributes take a value that is particular to the subclass in question. Each of these attribute values is defined in the LUO by means of a corresponding rule, which is described in Subsection 4.1.3.2.4. For this reason, these subclass-attribute associations have not been included here in order to avoid redundancy and for the sake of conciseness.

Table 42: Attributes of the Syntactic Level units within the LUO

CONCEPT	ATTRIBUTE	VALUES
Syntactic Unit	hasRank	{TOKEN, PHRASE, CLAUSE, SENTENCE}
Token	isSimple	BOOLEAN
Morphosyntactic Unit	isPunctuation	BOOLEAN
Word	hasGrammaticalCategory	{ADJECTIVE, ADPOSITION, ADVERB, ARTICLE, CONJUNCTION, INTERJECTION, NOUN, NUMERAL,

¹⁰⁰ This concept has not been introduced yet for the sake of clarity, since it is linked to the remaining concepts of this module just by means of an *ad hoc* relation shown below.

CONCEPT	ATTRIBUTE	VALUES
		PRONOUN-DETERMINER, RESIDUAL, UNIQUE, VERB}
Adjective	isCardinal	BOOLEAN
Adposition	isFused	BOOLEAN
Adverb	hasAdverbType	{DEGREE, PARTICLE, GENERAL, PRONOMINAL}
General Adverb	isWhAdverb	BOOLEAN
Pronominal Adverb	isWhAdverb	BOOLEAN
Wh-Adverb	hasWhAdverbType	{EXCLAMATORY, INTERROGATIVE, RELATIVE}
Article	hasArticleType	{DEFINITE, INDEFINITE, PARTITIVE}
Conjunction	isCoordinating	BOOLEAN
Coordinating Conjunction	hasCCType	{CORRELATIVE, INITIAL, NON-INITIAL, SIMPLE}
Simple Coordinating Conjunction	hasSimpleCCType	{ADVERSATIVE, COPULATIVE, DISJUNCTIVE}
Subordinating Conjunction	hasSCType	{ADVERBIAL, COMPARATIVE, FINITE, INFINITE, SUBSTANTIVE}
Noun	isProper	BOOLEAN
Common Noun	isCountable	BOOLEAN
Numeral	isCardinal	BOOLEAN
Numeral	isAdjective	BOOLEAN
Numeral	isPronoun	BOOLEAN
Numeral	isDeterminer	BOOLEAN
Pronoun/Determiner	isPronoun	BOOLEAN
Pronoun/Determiner	hasPDType	{DEMONSTRATIVE, EXCLAMATORY, INDEFINITE, INTERROGATIVE, NUMERAL, PARTITIVE, PERSONAL, POSSESSIVE, RECIPROCAL, REFLEXIVE, RELATIVE}
Numeral Pronoun/Determiner	isCardinal	BOOLEAN
Residual	hasResidualType	{ABBREVIATION, FOREIGN, FORMULA, SYMBOL, UNCLASSIFIED}
Verb	hasVerbType	{AUXILIARY, MAIN, SEMI-AUXILIARY}
Auxiliary Verb	isModal	BOOLEAN
Main Verb	isPredicative	BOOLEAN
Semi-Auxiliary Verb	isModal	BOOLEAN
Punctuation Mark	hasPunctMarkType	{SIMPLE-QUOTE, DOUBLE-QUOTE, PARENTHESIS, BRACKET, OTHER}
Phrase	hasPhraseType	{ADJECTIVE, ADVERB, NOUN, PREPOSITIONAL, VERB}
Clause	hasClauseType	{ADVERBIAL, COMPARATIVE, NOMINAL, RELATIVE}

4.1.3.2.3 The Syntactic Module *Ad Hoc* Relations

This subsection deals with the *ad hoc* relations bearing between syntactic units. They have been summarised in Table 43.

Table 43: Syntactic *ad hoc* relations in the LUO

SOURCE CONCEPT	AD HOC RELATION	TARGET CONCEPT
Token	<i>DependsSyntacticallyOn</i>	Token
Token	<i>CollocatesWith</i>	Token
Simple Token (Morphosyntactic Unit)	<i>hasLemma</i>	Lemma (Canonical Form)

4.1.3.2.4 The Syntactic Module Rules and Axioms

This subsection shows the rules that specify the semantics of the rest of the terms formalised in the syntactic module of the LUO (no axiom has been found to hold within this module). Most of them determine the values that the class attributes defined in the previous subsection take within a given concept. These values are presented, using a tabular notation, in Table 44.

Table 44: Rules associated to the attribute values of the LUO syntactic units

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SYN.001	Syntactic Unit	LinguisticUnitType	SYNTACTIC
R.LUO.SYN.002	Multiword Token	isMultiwordExpression	TRUE
R.LUO.SYN.003	Token	hasRank	TOKEN
R.LUO.SYN.004	Phrase	hasRank	PHRASE
R.LUO.SYN.005	Clause	hasRank	CLAUSE
R.LUO.SYN.006	Sentence	hasRank	SENTENCE
R.LUO.SYN.007	Morphosyntactic Unit (Simple Token)	LinguisticUnitType	{MORPHOLOGICAL, SYNTACTIC}
R.LUO.SYN.008	Morphosyntactic Unit (Simple Token)	isSimple	TRUE
R.LUO.SYN.009	Multiword Token	isSimple	FALSE
R.LUO.SYN.010	Punctuation Mark	isPunctuation	TRUE
R.LUO.SYN.011	Word	isPunctuation	FALSE
R.LUO.SYN.012	Adjective	hasGrammaticalCategory	ADJECTIVE
R.LUO.SYN.013	Adposition	hasGrammaticalCategory	ADPOSITION
R.LUO.SYN.014	Adverb	hasGrammaticalCategory	ADVERB
R.LUO.SYN.015	Article	hasGrammaticalCategory	ARTICLE
R.LUO.SYN.016	Conjunction	hasGrammaticalCategory	CONJUNCTION

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SYN.017	Interjection	hasGrammaticalCategory	INTERJECTION
R.LUO.SYN.018	Noun	hasGrammaticalCategory	NOUN
R.LUO.SYN.019	Numeral	hasGrammaticalCategory	NUMERAL
R.LUO.SYN.020	Pronoun/Determiner	hasGrammaticalCategory	PRONOUN-DETERMINER
R.LUO.SYN.021	Residual	hasGrammaticalCategory	RESIDUAL
R.LUO.SYN.022	Unique/Unassigned	hasGrammaticalCategory	UNIQUE
R.LUO.SYN.023	Verb	hasGrammaticalCategory	VERB
R.LUO.SYN.024	Cardinal Adjective	isCardinal	TRUE
R.LUO.SYN.025	Ordinal Adjective	isCardinal	FALSE
R.LUO.SYN.026	Fused Prep-At (Contraction)	isFused	TRUE
R.LUO.SYN.027	Proper Adposition	isFused	FALSE
R.LUO.SYN.028	Degree Adverb	hasAdverbType	DEGREE
R.LUO.SYN.029	Particle Adverb	hasAdverbType	PARTICLE
R.LUO.SYN.030	General Adverb	hasAdverbType	GENERAL
R.LUO.SYN.031	Pronominal Adverb	hasAdverbType	PRONOMINAL
R.LUO.SYN.032	Wh-Adverb	isWhAdverb	TRUE
R.LUO.SYN.033	Non Wh-Adverb	isWhAdverb	FALSE
R.LUO.SYN.034	Exclamatory Adverb	hasWhAdverbType	EXCLAMATORY
R.LUO.SYN.035	Interrogative Adverb	hasWhAdverbType	INTERROGATIVE
R.LUO.SYN.036	Relative Adverb	hasWhAdverbType	RELATIVE
R.LUO.SYN.037	Definite Article	hasArticleType	DEFINITE
R.LUO.SYN.038	Indefinite Article	hasArticleType	INDEFINITE
R.LUO.SYN.039	Partitive Article	hasArticleType	PARTITIVE
R.LUO.SYN.040	Coordinating Conjunction	isCoordinating	TRUE
R.LUO.SYN.041	Subordinating Conjunction	isCoordinating	FALSE
R.LUO.SYN.042	Correlative Coordinating Conjunction	hasCCType	CORRELATIVE
R.LUO.SYN.043	Initial Coordinating Conjunction	hasCCType	INITIAL
R.LUO.SYN.044	Non-Initial Coordinating Conjunction	hasCCType	NON-INITIAL
R.LUO.SYN.045	Simple Coordinating Conjunction	hasCCType	SIMPLE
R.LUO.SYN.046	Adversative Conjunction	hasSimpleCCType	ADVERSATIVE
R.LUO.SYN.047	Copulative Conjunction	hasSimpleCCType	COPULATIVE
R.LUO.SYN.048	Disjunctive Conjunction	hasSimpleCCType	DISJUNCTIVE
R.LUO.SYN.049	Adverbial Subordinating Conjunction	hasSCType	ADVERBIAL
R.LUO.SYN.050	Comparative Subordinating Conjunction	hasSCType	COMPARATIVE

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SYN.051	With Finite Subordinating Conjunction	hasSCType	FINITE
R.LUO.SYN.052	With Infinite Subordinating Conjunction	hasSCType	INFINITE
R.LUO.SYN.053	Substantive Subordinating Conjunction	hasSCType	SUBSTANTIVE
R.LUO.SYN.054	Proper Noun	isProper	TRUE
R.LUO.SYN.055	Common Noun	isProper	FALSE
R.LUO.SYN.056	Countable Noun	isCountable	TRUE
R.LUO.SYN.057	Mass Noun	isCountable	FALSE
R.LUO.SYN.058	Cardinal	isCardinal	TRUE
R.LUO.SYN.059	Ordinal	isCardinal	FALSE
R.LUO.SYN.060	Cardinal Adjective	isAdjective	TRUE
R.LUO.SYN.061	Ordinal Adjective	isAdjective	TRUE
R.LUO.SYN.062	Cardinal Pronoun/Determiner	isAdjective	FALSE
R.LUO.SYN.063	Ordinal Pronoun/Determiner	isAdjective	FALSE
R.LUO.SYN.064	Cardinal Determiner	isAdjective	FALSE
R.LUO.SYN.065	Cardinal Pronoun	isAdjective	FALSE
R.LUO.SYN.066	Ordinal Determiner	isAdjective	FALSE
R.LUO.SYN.067	Ordinal Pronoun	isAdjective	FALSE
R.LUO.SYN.068	Cardinal Pronoun	isPronoun	TRUE
R.LUO.SYN.069	Ordinal Pronoun	isPronoun	TRUE
R.LUO.SYN.070	Cardinal Adjective	isPronoun	FALSE
R.LUO.SYN.071	Ordinal Adjective	isPronoun	FALSE
R.LUO.SYN.072	Cardinal Determiner	isPronoun	FALSE
R.LUO.SYN.073	Ordinal Determiner	isPronoun	FALSE
R.LUO.SYN.074	Cardinal Determiner	isDeterminer	TRUE
R.LUO.SYN.075	Ordinal Determiner	isDeterminer	TRUE
R.LUO.SYN.076	Cardinal Adjective	isDeterminer	FALSE
R.LUO.SYN.077	Ordinal Adjective	isDeterminer	FALSE
R.LUO.SYN.078	Cardinal Pronoun	isDeterminer	FALSE
R.LUO.SYN.079	Ordinal Pronoun	isDeterminer	FALSE
R.LUO.SYN.080	Demonstrative Pronoun	isPronoun	TRUE
R.LUO.SYN.081	Exclamatory Pronoun	isPronoun	TRUE
R.LUO.SYN.082	Indefinite Pronoun	isPronoun	TRUE
R.LUO.SYN.083	Interrogative Pronoun	isPronoun	TRUE
R.LUO.SYN.084	Personal Pronoun	isPronoun	TRUE
R.LUO.SYN.085	Possessive Pronoun	isPronoun	TRUE
R.LUO.SYN.086	Reciprocal Pronoun	isPronoun	TRUE
R.LUO.SYN.087	Reflexive Pronoun	isPronoun	TRUE

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SYN.088	Relative Pronoun	isPronoun	TRUE
R.LUO.SYN.089	Demonstrative Determiner	isPronoun	FALSE
R.LUO.SYN.090	Indefinite Determiner	isPronoun	FALSE
R.LUO.SYN.091	Interrogative Determiner	isPronoun	FALSE
R.LUO.SYN.092	Partitive Determiner	isPronoun	FALSE
R.LUO.SYN.093	Possessive Determiner	isPronoun	FALSE
R.LUO.SYN.094	Relative Determiner	isPronoun	FALSE
R.LUO.SYN.095	Demonstrative Pronoun/Determiner	hasPDType	DEMONSTRATIVE
R.LUO.SYN.096	Exclamatory Pronoun	hasPDType	EXCLAMATORY
R.LUO.SYN.097	Indefinite Pronoun/Determiner	hasPDType	INDEFINITE
R.LUO.SYN.098	Interrogative Pronoun/Determiner	hasPDType	INTERROGATIVE
R.LUO.SYN.099	Numeral Pronoun/Determiner	hasPDType	NUMERAL
R.LUO.SYN.100	Partitive Determiner	hasPDType	PARTITIVE
R.LUO.SYN.101	Personal Pronoun	hasPDType	PERSONAL
R.LUO.SYN.102	Possessive Pronoun/Determiner	hasPDType	POSSESSIVE
R.LUO.SYN.103	Reciprocal Pronoun	hasPDType	RECIPROCAL
R.LUO.SYN.104	Reflexive Pronoun	hasPDType	REFLEXIVE
R.LUO.SYN.105	Relative Pronoun/Determiner	hasPDType	RELATIVE
R.LUO.SYN.106	Abbreviation	hasResidualType	ABBREVIATION
R.LUO.SYN.107	Foreign Word	hasResidualType	FOREIGN
R.LUO.SYN.108	Formula	hasResidualType	FORMULA
R.LUO.SYN.109	Symbol	hasResidualType	SYMBOL
R.LUO.SYN.110	Unclassified Residual	hasResidualType	UNCLASSIFIED
R.LUO.SYN.111	Auxiliary Verb	hasVerbType	AUXILIARY
R.LUO.SYN.112	Main Verb	hasVerbType	MAIN
R.LUO.SYN.113	Semi-Auxiliary Verb	hasVerbType	SEMI-AUXILIARY
R.LUO.SYN.114	Modal Auxiliary Verb	isModal	TRUE
R.LUO.SYN.115	Primary Auxiliary Verb	isModal	FALSE
R.LUO.SYN.116	Copular Verb	isPredicative	FALSE
R.LUO.SYN.117	Predicative Verb	isPredicative	TRUE
R.LUO.SYN.118	Aspectual Semi-Auxiliary Verb	isModal	FALSE
R.LUO.SYN.119	Modal Semi-Auxiliary Verb	isModal	TRUE
R.LUO.SYN.120	Simple Quotation Mark	hasPunctMarkType	SIMPLE-QUOTE
R.LUO.SYN.121	Double Quotation Mark	hasPunctMarkType	DOUBLE-QUOTE
R.LUO.SYN.122	Parenthesis	hasPunctMarkType	PARENTHESIS
R.LUO.SYN.123	Bracket	hasPunctMarkType	BRACKET
R.LUO.SYN.124	Other Punctuation Mark	hasPunctMarkType	OTHER

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SYN.125	Adjective Phrase	hasPhraseType	ADJECTIVE
R.LUO.SYN.126	Adverb Phrase	hasPhraseType	ADVERB
R.LUO.SYN.127	Prepositional Phrase	hasPhraseType	PREPOSITIONAL
R.LUO.SYN.128	Noun Phrase	hasPhraseType	NOUN
R.LUO.SYN.129	Verb Phrase	hasPhraseType	VERB
R.LUO.SYN.130	Adverbial Clause	hasClauseType	ADVERBIAL
R.LUO.SYN.131	Comparative Clause	hasClauseType	COMPARATIVE
R.LUO.SYN.132	Nominal Clause	hasClauseType	NOMINAL
R.LUO.SYN.133	Relative Clause	hasClauseType	RELATIVE

Hence, the concepts of the syntactic module of the LUO, which formalise morphosyntactic and syntactic units, have been presented thus far, together with their associated attributes, *ad hoc* relations, rules and axioms. To continue the presentation of the different units of the LUO, its Semantic Level units and their corresponding attributes, *ad hoc* relations, rules and axioms, are introduced in the next subsection.

4.1.3.3 THE SEMANTIC MODULE

This module of the LUO formalises the linguistic units belonging to the Semantic Level (*i.e.*, the semantic units). The concepts representing the semantic units and the taxonomical relations holding between them are presented in the following subsection (4.1.3.3.1). The *attributes* characterizing them are included in Subsection 4.1.3.3.2. The *ad hoc relations* holding between these semantic elements are discussed in Subsection 4.1.3.3.3. The *axioms* and *rules* that constrain and limit their semantics are shown in Subsection 4.1.3.3.4.

4.1.3.3.1 The Semantic Module Concepts and Taxonomy

The concepts of the LUO representing semantic units, as well as the taxonomical relations that hold (i) between these concepts and (ii) between them and the LUO top-level concepts are all summarised from Table 45 to Table 59.

Table 45 presents the semantic top-level concepts of the LUO. They represent the main subclasses of Semantic Unit that have been formalised in OntoTag’s ontologies. As shown in this table, the

class `Semantic Unit` has been subclassified into the following classes: `Sense`, `Propositional Component` and `Other Semantic Unit`¹⁰¹.

Some considerations need to be made about this subclassification of `Semantic Unit`. First, the class `Other Semantic Unit` was included in the ontology (1) for the sake of completeness; (2) in order to facilitate the expansion of this part of the ontology, by adding any other semantic units that might be identified in the future as subclasses of this concept; and (3) to assure that an *Exhaustive-Decomposition* was attained. Second, it may seem that the classes `Sense` and `Propositional Component` are redundant and, thus, one of them could be removed from the ontology. In fact, both of them formalise certain overlapping phenomena, but `Propositional Component` formalises some particular phenomena that are not entailed by the concept `Sense`.

Consider, for example, the sentence ‘This morning I saw John in the park’. On the one hand, the `Sense` units involved in this sentence (*i.e.*, the synsets, for example) would be those associated to (a) ‘morning’, (b) ‘see’ (the Lemma associated to ‘saw’) and (c) ‘park’. On the other hand, the `Propositional Component` units involved in this Sentence would be those associated to: (1) ‘this morning’, which requires a time deictic (pragmatic) resolution, since it refers to a concrete time interval (a concrete morning of a concrete day); (2) ‘I’, which also requires a person deictic (pragmatic) resolution, in order to identify the `Agent` of the `Action` meant by ‘saw’ in the discourse; (3) ‘see’ (*id.*); (4) ‘John’, which is a `Named Entity` that might require some person deictic resolution as well; and (5) ‘in the park’, which details the `Location` of the `Action` entailed by ‘saw’, which is clearly different from and contrasts with, for example, ‘near the park’ or ‘out of the park’, and which also requires a spatial deictic (pragmatic) resolution, in order to determine the concrete ‘park’ meant by the `Agent` of the `Proposition` and its concrete spatial location.

As shown in the example above, (1) there are more `Propositional Component` units than `Sense` units, that is, there are cases in which a `Propositional Component` unit has no correspondence with any `Sense` unit; (2) only in the case of ‘saw’ it can be considered to exist a full correspondence between the `Sense` units and the `Propositional Component` units; and (3) in the rest of cases, where some degree of correspondence can be observed, the `Propositional Component` unit has a more particularised and concrete meaning than the one entailed by the `Sense` unit. Therefore, both types of units need to be formalised (independently) within the LUO.

¹⁰¹ Like in previous cases, this last concept has been added for the sake of completeness and in order to facilitate the expansion of this part of the ontology, by adding any other semantic units that might be identified in the future as subclasses of this concept.

Only one **Subclass-Of Sense** has been identified so far, namely *Synset*. It has been included as well in Table 45. It formalises the idea of *synset*, crucial in both WordNet and EuroWordNet, which are the *de facto* standards for the computational representation of senses nowadays.

Table 45: The subclasses of Semantic Unit in the LUO and their taxonomical relations

Semantic Unit	Sense	Synset
	Propositional Component	[Continued in Table 46]
	Other Semantic Unit	

As for Table 46, it presents the main **subclasses of Propositional Component**, as well as a simplified representation of the taxonomical relations holding between them. The main concepts in this table (*Entity*, *Action*, and *Quality*) have been extracted from Lyons (1977). For this author, these main concepts represent the ontological basis of Semantics and Grammar. Some *synonyms* for *Action* (*Event*, *State Of Affairs* and *Process*) have been included in this module of the LUO as well. These *synonyms* are the terms used in other linguistic grammars and theories to refer to the same concept.

Table 46: The subclasses of Propositional Component in the LUO and the taxonomical relations holding between them

Propositional Component	Entity ¹⁰²		[Continued in Table 47]
	Action (Event, Process, State Of Affairs, SoA)		[Continued in Table 57]
	Quality	Property	[Continued in Table 58]
		Circumstance	[Continued in Table 59]

On the one hand, *State Of Affairs* shows the correspondence of this concept with its related attributes within the Linguistic Attribute Ontology (the LAO – see Subsection 4.1.4.1.3 for details), extracted from Dik (1989). On the other hand, *Process* shows the correspondence of this concept with its subclasses, derived from Halliday (1994; 1996). The **subclassification of Quality** into *Property* and *Circumstance* is based on the fact that there are two different types of qualities: *entity qualities* (that is, *properties*) and *event qualities* (that is, *circumstances*).

Regarding Table 47, it shows the **main subclasses of Entity**, namely *Named Entity* and *Generic Entity*, as well as the (main) subclasses of the latter. Besides, this table also shows a simplified representation of the taxonomical relations holding between all these classes. The subclass of *Entity* referred to as *Named Entity* has been included here for the sake of completeness (with respect to the existing annotation schemes up to date). In the present model, any *Named Entity* is an

¹⁰² This concept is referred to as *Participant* in Halliday (1994; 1996).

Instance-Of a *Subclass-Of* Generic Entity, and is annotated as such in the corresponding Semantic Annotation Layer (*i.e.*, the Instance Semantic Annotation Layer – see Subsections 4.2.5.1.4 and 4.3.1.3.2 for details). The subclasses of Generic Entity are, basically, the subclasses of Entity in a top-level ontology developed for the EuroWordNet elements (synsets) within the SIMPLE (2000) project. However, this top-level ontology was (slightly) adapted before being formalised and included in the present module of the LUO.

Table 47: The subclasses of Entity in the LUO and the taxonomical relations holding between them

Entity	Named Entity		
	Generic Entity	Concrete Entity	[Continued in Table 48]
		Abstract Entity	[Continued in Table 55]

With respect to Table 48, it includes the **main subclasses of Concrete Entity**. As commented, they were extracted from the EuroWordNet top-level ontology aforementioned.

Table 48: The subclasses of Concrete Entity in the LUO (taken from SIMPLE (2000))

Concrete Entity	Location	[Continued in Table 49]
	Material	[Continued in Table 50]
	Artifact	[Continued in Table 51]
	Food	[Continued in Table 52]
	Physical Object	
	Organic Object	
	Living Entity	[Continued in Table 53]
	Substance	[Continued in Table 54]

As for Table 49, Table 50, Table 51, Table 52, Table 53 and Table 54, they present the different subclasses of the main subclasses of Concrete Entity included in this module of the LUO, that is, the subclasses of Location, Material, Artifact, Food, Living Entity and Substance, respectively. They come from the EuroWordNet top-level ontology taken from SIMPLE (2000) as well. No associated subclassification for Physical Object or Organic Object was included in SIMPLE (2000). Thus, these concepts were not further subclassified in the LUO either.

Table 49: The subclasses of Location in the LUO and the taxonomical relations holding between them (taken from SIMPLE (2000))

Location	3D Location		
	Geopolitical Location		
	Area		
	Opening		
	Artifactual Location ¹⁰³	Building	
		Artifactual Area	
	Other Location		

Table 50: The subclasses of Material in the LUO (taken from SIMPLE (2000))

Material	Non-Artifactual Material
	Artifactual Material

Table 51: The subclasses of Artifact in the LUO (taken from SIMPLE (2000))

Artifact	Artifactual Location
	Artifactual Material ¹⁰⁴
	Furniture
	Clothing
	Container
	Artwork
	Instrument
	Money
	Vehicle
	Semiotic Artifact
	Artifactual Food
	Other Artifact

Table 52: The subclasses of Food in the LUO (taken from SIMPLE (2000))

Food	Artifactual Food ¹⁰⁵
	Non-Artifactual Food
	Flavouring

¹⁰³ This concept is also a *Subclass-Of* Artifact.

¹⁰⁴ Artifactual Material is a *Subclass-Of* Material as well.

¹⁰⁵ This concept is also a *Subclass-Of* Artifact.

Table 53: The subclasses of Living Entity in the LUO (taken from SIMPLE (2000))

Living Entity	Animal	Earth Animal	
		Air Animal	
		Water Animal	
	Human	People	
		Role	Ideo
			Kinship
			Social Status
		Agent Of Activity	Agent Of Temporary Activity
			Agent Of Persistent Activity
	Professional		
	Vegetal Entity	Plant	
		Flower	
		Fruit	
	Micro-Organism		

Table 54: The subclasses of Substance in the LUO (taken from SIMPLE (2000))

Substance	Natural Substance	
	Food Substance ¹⁰⁶	
	Drink	Artifactual Drink ¹⁰⁷

Concerning Table 55, it contains the **main subclasses of Abstract Entity** in the adapted EuroWordNet top-level ontology taken from SIMPLE (2000).

Table 55: The subclasses of Abstract Entity in the LUO and the taxonomical relations holding between them

Abstract Entity	Domain	[Continued in Table 56]
	Time	
	Moral Standard	
	Cognitive Fact	
	Movement Of Thought	
	Institution	
	Convention	

Table 56, in turn, summarises the **main subclasses of Domain**, according to the lexical subclassification of this concept found in SIMPLE (2000).

¹⁰⁶ This concept is also a *Subclass-Of* Food.

¹⁰⁷ This concept is also a *Subclass-Of* Artifact.

Table 56: The main subclasses of *Domain* in the LUO (taken from SIMPLE (2000))

Domain	General Domain	
	Food Domain	
	Agriculture-Fishing-Forestry	
	Business Domain	
	Industry Domain	Service Industry Domain
		Craft Industry Domain
		Manufacturing Industry Domain
	Construction Domain	
	Politics-Government	
	Sciences	
	Health-Medicine	
	Military Domain	
	Home-Garden	
	Education Domain	
	Sports-Leisure	
	Arts	
	Religion Domain	
	Transport Domain	
	Law Domain	
	Themes	

Regarding Table 57, it shows the **main subclasses of Action**, as well as a simplified representation of the taxonomical relations holding between them. This subclassification of *Action* has been extracted from Halliday (1994; 1996), instead of extracting it from the *Event* subclassification included in SIMPLE (2000). When compared to the SIMPLE subclassification of *Event*, Halliday’s subclassification of processes seems more psychologically motivated (and, hence, more meaning-oriented, or, equivalently, more suitable as for its use within the Semantic Web). Besides, there is a clear correspondence between Halliday’s subclassification of processes and the semantic roles that semantic components (or participants) play in propositions. This is why the subclassification in SIMPLE (2000) was set aside in this case.

Table 57: The Action subclasses and the taxonomical relations holding between them

Action (Event, Process, State Of Affairs, SoA) ¹⁰⁸	Material Process	Dispositive Material Process	
		Creative Material Process	
	Mental Process	Perception Process	
		Cognition Process	
		Affection Process	
	Relational Process	Intensive Process	Attributive Intensive Process
			Identifying Intensive Process
		Circumstantial Process	Attributive Circumstantial Process
			Identifying Circumstantial Process
		Possessive Process	Attributive Possessive Process
			Identifying Possessive Process
	Other Process	Existential Process	
		Behavioural Process	
		Verbal Process	

Concerning Table 58, it presents the **main subclasses of Property**, as well as a simplified representation of the taxonomical relations holding between them. The concepts in this table have been derived from the top-ontology for adjectives included in SIMPLE (2000). In the original source, it is used as an abstraction of the meaning of adjectives. It has been slightly adapted here so as to represent the meaning of all kind of properties.

Table 58: The Property subclasses and the taxonomical relations holding between them

Property	Intensional Property	Modality	
		Temporality	
		Emotiveness	
		Manner Property	
		Object-Relatedness	
		Emphasis	
	Extensional Property	Physical Property	Body Property
			Perception Property
			Movement Property
			Space Property

¹⁰⁸ As mentioned above, this concept is called Action in the original source. The synonyms Event, Process and State Of Affairs have been included in the ontology in order to show its connection with the rest of sources used to build this module of the LUO.

			Substance Property
			Other Physical Property
		Psychological Property	Experiential Property
			Psychological State Property
			Cognition Property
			Attitude Salience Property
			Attitude Evaluation Property
			Other Psychological Property
Political Property			
Legal Property			
Nationality			
Military Property			
Economical Property			
		Temporal Property	
		Intensifying Property	Frequency Intensification
			Power Intensification
			Other Intensification
		Relational Property	

Finally, regarding Table 59, it includes the **(main) subclasses of Circumstance**, as well as a simplified representation of the taxonomical relations holding between them. The concepts in this table have been extracted from Halliday (1994; 1996). They stand for the *circumstantial element hierarchy* described in the cited work. The reason for choosing this hierarchy is its broad and truly semantic classification of circumstances.

Table 59: The Circumstance subclasses and the taxonomical relations holding between them (taken from Halliday (1994; 1996))

Circumstance	Extent Circumstance	Distance Circumstance
		Duration Circumstance
	Cause	Purpose Circumstance
		Reason Circumstance
		Behalf Circumstance
	Location Circumstance	Place Circumstance
		Time Circumstance
	Contingency	Condition Circumstance
		Concession Circumstance
		Default Circumstance
	Role	Guise Circumstance
		Product Circumstance

	Angle	
	Manner	Means Circumstance
		Comparison Circumstance
		Quality Circumstance
	Matter	
	Accompaniment	Addition Circumstance
		Comitacion Circumstance

As far as the taxonomical relations that bear between all these latter semantic units, they have been formalised according to the same criteria applied in previous subsections. There are *Partitions*, *Disjoint-Decompositions*, *Exhaustive-Decompositions* and simple *Subclass-Of* relations. However, a complete discussion about their determination is not included here for the sake of space.

Up to this point, the semantic units contemplated in the LUO and their corresponding taxonomical relations have already been presented. There remains to present as well their attributes, *ad hoc* relations and their associated rules and axioms. This will be done in the next subsections.

4.1.3.3.2 The Semantic Module Attributes

The attributes associated to the semantic units of the LUO are presented in this subsection. All of them are *class attributes*. These attributes are included in Table 60, which shows, for each attribute, the unit to which it is ascribed and the values that it can take. Some of the attributes associated to the concepts in this table are associated to its subclasses as well. In these subclasses, these attributes take a value that is particular to that subclass. Therefore, it is defined by means of a corresponding rule in Subsection 4.1.3.3.4. For this reason, these subclass-attribute associations have not been included here in order to avoid redundancy and for the sake of conciseness.

Table 60: Attributes of the semantic units within the LUO

CONCEPT	ATTRIBUTE	VALUES
Semantic Unit	hasLexicalLevel	{SENSE, PROP-COMP, OTHER}
Propositional Component	hasPropCompType	{ENTITY, ACTION, QUALITY}
Quality	hasQualityType	{PROPERTY, CIRCUMSTANCE}
Property	isIntensional	BOOLEAN
Intensional Property	hasIntensionalPropType	{MODALITY, TEMPORALITY, EMOTIVENESS, MANNER, OBJECT-RELATED, EMPHASIS}
Extensional Property	hasExtensionalPropType	{PHYSICAL, PSYCHOLOGICAL, SOCIAL, TEMPORAL, INTENSIFYING, RELATIONAL}
Physical	hasPhysicalPropType	{BODY, PERCEPTION, MOVEMENT, SPACE, SUBSTANCE,

CONCEPT	ATTRIBUTE	VALUES
Property		OTHER}
Psychological Property	hasPsychologicalPropType	{EXPERIENTIAL, PSYCHOLOGICAL, COGNITIVE, ATTITUDE_SALIENCE, ATTITUDE_EVAL, OTHER}
Social Property	hasSocialPropType	{RELIGIOUS, POLITICAL, LEGAL, MILITARY, ECONOMICAL, OTHER}
Intensifying Property	hasIntensifyingPropType	{FREQUENCY, POWER, OTHER}
Circumstance	hasPropertyType	{EXTENT, CAUSE, LOCATION, CONTINGENCY, ROLE, ANGLE, MANNER, MATTER, ACCOMPANIMENT}
Extent Circumstance	hasExtentType	{DISTANCE, DURATION}
Cause	hasCauseType	{PURPOSE, REASON, BEHALF}
Location	hasLocationType	{PLACE, TIME}
Contingency	hasContingencyType	{CONDITION, CONCESSION, DEFAULT}
Role	hasRoleType	{GUISE, PRODUCT}
Manner	hasMannerType	{MEANS, COMPARISON, QUALITY}
Accompaniment	hasAccompanimentType	{ADDITION, COMITATION}
Entity	isNamedEntity	BOOLEAN
Generic Entity	isConcrete	BOOLEAN
Concrete Entity	hasConcreteEntityType	{LOCATION, MATERIAL, ARTIFACT, FOOD, PHYSICAL_OBJECT, ORGANIC_OBJECT, LIVING_ENTITY, SUBSTANCE}
Location	hasLocationType	{3D, GEOPOLITICAL, AREA, OPENING, ARTIFACTUAL_LOCATION, OTHER}
Material	IsArtifactual	BOOLEAN
Artifact	hasArtifactType	{ARTIFACTUAL_MATERIAL, FURNITURE, CLOTHING, CONTAINER, ARTWORK, INSTRUMENT, MONEY, VEHICLE, SEMIOTIC_ARTIFACT, ARTIFACTUAL_FOOD, OTHER}
Food	hasFoodType	{ARTIFACTUAL, NON-ARTIFACTUAL, FLAVOURING}
Living Entity	hasLivingEntityType	{ANIMAL, HUMAN, VEGETAL, MICRO}
Animal	hasAnimalType	{EARTH, AIR, WATER}
Human	hasHumanType	{PEOPLE, ROLE, AGENT, PROFESSIONAL}
Role	hasRoleType	{IDEO, KINSHIP, SOCIAL_STATUS}
Agent	performsPersistentActivity	BOOLEAN
Vegetal	hasVegetalType	{PLANT, FLOWER, FRUIT}
Substance	hasSubstanceType	{NATURAL, FOOD, DRINK}
Drink	isArtifactual	BOOLEAN
Abstract Entity	hasAbstractEntityType	{DOMAIN, TIME, MORAL_STANDARD, COGNITIVE_FACT, MOVEMENT_OF_THOUGHT, INSTITUTION, CONVENTION}
Domain	hasDomainType	{GENERAL, FOOD, AGRIC-FISH-FOREST, BUSINESS, INDUSTRY, CONSTRUCTION, POLITICS, SCIENCES,

CONCEPT	ATTRIBUTE	VALUES
		HEALTH, MILITARY, HOME, EDUCATION, SPORTS, ARTS, RELIGION, TRANSPORT, LAW, THEMES}
Industry Domain	hasIndustryDomain	{SERVICE, CRAFT, MANUFACTURING}
Action (Event, Process)	hasActionType	{MATERIAL, MENTAL, RELATIONAL, OTHER}
Material Process	hasMaterialProcessType	{DISPOSITIVE, CREATIVE}
Mental Process	hasMentalProcessType	{PERCEPTION, COGNITION, AFFECTION}
Relational Process	hasRelationalProcessType	{INTENSIVE, CIRCUMSTANTIAL, POSSESSIVE}
Intensive Process	isAttributiveProcess	BOOLEAN
Circumstantial Process	isAttributiveProcess	BOOLEAN
Possessive Process	isAttributiveProcess	BOOLEAN
Other Process	hasOtherProcessType	{EXISTENTIAL, BEHAVIOURAL, VERBAL}

4.1.3.3.3 The Semantic Module *Ad Hoc* Relations

This subsection deals with the *ad hoc* relations bearing between semantic units. They have been summarised in Table 61.

Table 61: Semantic *ad hoc* relations in the LUO

SOURCE CONCEPT	AD HOC RELATION	TARGET CONCEPT
Semantic Unit	<i>isSemanticConstituentOf</i>	Semantic Unit
Semantic Unit	<i>DependsSemanticallyOn</i>	Semantic Unit
Semantic Unit	<i>hasHyponym</i>	Semantic Unit
Semantic Unit	<i>hasHyperonym</i>	Semantic Unit
Semantic Unit	<i>hasHolonym</i>	Semantic Unit
Semantic Unit	<i>hasMeronym</i>	Semantic Unit
Semantic Unit	<i>hasAntonym</i>	Semantic Unit
Noun	<i>hasSense</i>	Sense
Verb	<i>hasSense</i>	Sense
Adjective	<i>hasSense</i>	Sense
Adverb	<i>hasSense</i>	Sense

SOURCE CONCEPT	AD HOC RELATION	TARGET CONCEPT
Semantic Unit	<i>hasDomain</i>	Domain
Named Entity	<i>RefersTo</i> ¹⁰⁹	Generic Entity

4.1.3.3.4 The Semantic Module Rules and Axioms

This subsection shows the rules that specify the semantics of the rest of the terms formalised in the semantic module of the LUO (no axiom has been found to hold within this module either). Most of them determine the values that the class attributes defined in the previous subsection take within a given concept. These values are presented, using a tabular notation, in Table 62.

Table 62: Rules associated to the attribute values of the LUO semantic units

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SEM.001	Sense	hasSemUnitType	SENSE
R.LUO.SEM.002	Propositional Component	hasSemUnitType	PROP-COMP
R.LUO.SEM.003	Other Simple Lexical Meaning Unit	hasSemUnitType	OTHER
R.LUO.SEM.004	Entity	hasPropCompType	ENTITY
R.LUO.SEM.005	Action (Event, Process, State Of Affairs, SoA)	hasPropCompType	ACTION
R.LUO.SEM.006	Quality	hasPropCompType	QUALITY
R.LUO.SEM.007	Property	hasQualityType	PROPERTY
R.LUO.SEM.008	Circumstance	hasQualityType	CIRCUMSTANCE
R.LUO.SEM.009	Intensional Property	isIntensional	TRUE
R.LUO.SEM.010	Extensional Property	isIntensional	FALSE
R.LUO.SEM.011	Modality	hasIntensionalPropType	MODALITY
R.LUO.SEM.012	Temporality	hasIntensionalPropType	TEMPORALITY
R.LUO.SEM.013	Emotiveness	hasIntensionalPropType	EMOTIVENESS
R.LUO.SEM.014	Manner Property	hasIntensionalPropType	MANNER
R.LUO.SEM.015	Object-Relatedness	hasIntensionalPropType	OBJECT-RELATED
R.LUO.SEM.016	Emphasis	hasIntensionalPropType	EMPHASIS
R.LUO.SEM.017	Physical Property	hasExtensionalPropType	PHYSICAL
R.LUO.SEM.018	Psychological Property	hasExtensionalPropType	PSYCHOLOGICAL
R.LUO.SEM.019	Social Property	hasExtensionalPropType	SOCIAL

¹⁰⁹ Every instance of the Named Entity class is a linguistic expression that realises an *Instance-Of* the Generic Entity class or, more concretely, of any of its subclasses. This fact has been formalised in the LUO by means of the present *ad hoc* relation.

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SEM.020	Temporal Property	hasExtensionalPropType	TEMPORAL
R.LUO.SEM.021	Intensifying Property	hasExtensionalPropType	INTENSIFYING
R.LUO.SEM.022	Relational Property	hasExtensionalPropType	RELATIONAL
R.LUO.SEM.023	Body Property	hasPhysicalPropType	BODY
R.LUO.SEM.024	Perception Property	hasPhysicalPropType	PERCEPTION
R.LUO.SEM.025	Movement Property	hasPhysicalPropType	MOVEMENT
R.LUO.SEM.026	Space Property	hasPhysicalPropType	SPACE
R.LUO.SEM.027	Substance Property	hasPhysicalPropType	SUBSTANCE
R.LUO.SEM.028	Other Physical Property	hasPhysicalPropType	OTHER
R.LUO.SEM.029	Experiential Property	hasPsychologicalPropType	EXPERIENTIAL
R.LUO.SEM.030	Psychological State Property	hasPsychologicalPropType	PSYCHOLOGICAL
R.LUO.SEM.031	Cognition Property	hasPsychologicalPropType	COGNITIVE
R.LUO.SEM.032	Attitude Salience Property	hasPsychologicalPropType	ATTITUDE_SALIENCE
R.LUO.SEM.033	Attitude Evaluation Property	hasPsychologicalPropType	ATTITUDE_EVAL
R.LUO.SEM.034	Other Psychological Property	hasPsychologicalPropType	OTHER
R.LUO.SEM.035	Religious Property	hasSocialPropType	RELIGIOUS
R.LUO.SEM.036	Political Property	hasSocialPropType	POLITICAL
R.LUO.SEM.037	Legal Property	hasSocialPropType	LEGAL
R.LUO.SEM.038	Military Property	hasSocialPropType	MILITARY
R.LUO.SEM.039	Economical Property	hasSocialPropType	ECONOMICAL
R.LUO.SEM.040	Other Social Property	hasSocialPropType	OTHER
R.LUO.SEM.041	Frequency Intensification	hasIntensifyingPropType	FREQUENCY
R.LUO.SEM.042	Power Intensification	hasIntensifyingPropType	POWER
R.LUO.SEM.043	Other Intensification	hasIntensifyingPropType	OTHER
R.LUO.SEM.044	Extent Circumstance	hasPropertyType	EXTENT
R.LUO.SEM.045	Cause	hasPropertyType	CAUSE
R.LUO.SEM.046	Location	hasPropertyType	LOCATION
R.LUO.SEM.047	Contingency	hasPropertyType	CONTINGENCY
R.LUO.SEM.048	Role	hasPropertyType	ROLE
R.LUO.SEM.049	Angle	hasPropertyType	ANGLE
R.LUO.SEM.050	Manner	hasPropertyType	MANNER
R.LUO.SEM.051	Matter	hasPropertyType	MATTER
R.LUO.SEM.052	Accompaniment	hasPropertyType	ACCOMPANIMENT

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SEM.053	Distance Circumstance	hasExtentType	DISTANCE
R.LUO.SEM.054	Duration Circumstance	hasExtentType	DURATION
R.LUO.SEM.055	Purpose Circumstance	hasCauseType	PURPOSE
R.LUO.SEM.056	Reason Circumstance	hasCauseType	REASON
R.LUO.SEM.057	Behalf Circumstance	hasCauseType	BEHALF
R.LUO.SEM.058	Place Circumstance	hasLocationType	PLACE
R.LUO.SEM.059	Time Circumstance	hasLocationType	TIME
R.LUO.SEM.060	Condition Circumstance	hasContingencyType	CONDITION
R.LUO.SEM.061	Concession Circumstance	hasContingencyType	CONCESSION
R.LUO.SEM.062	Default Circumstance	hasContingencyType	DEFAULT
R.LUO.SEM.063	Guise Circumstance	hasRoleType	GUISE
R.LUO.SEM.064	Product Circumstance	hasRoleType	PRODUCT
R.LUO.SEM.065	Means Circumstance	hasMannerType	MEANS
R.LUO.SEM.066	Comparison Circumstance	hasMannerType	COMPARISON
R.LUO.SEM.067	Quality Circumstance	hasMannerType	QUALITY
R.LUO.SEM.068	Addition Circumstance	hasAccompanimentType	ADDITION
R.LUO.SEM.069	Comitation Circumstance	hasAccompanimentType	COMITATION
R.LUO.SEM.070	Named Entity	isNamedEntity	TRUE
R.LUO.SEM.071	Generic Entity	isNamedEntity	FALSE
R.LUO.SEM.072	Concrete Entity	isConcrete	TRUE
R.LUO.SEM.073	Abstract Entity	isConcrete	FALSE
R.LUO.SEM.074	Location	hasConcreteEntityType	LOCATION
R.LUO.SEM.075	Material	hasConcreteEntityType	MATERIAL
R.LUO.SEM.076	Artifact	hasConcreteEntityType	ARTIFACT
R.LUO.SEM.077	Food	hasConcreteEntityType	FOOD
R.LUO.SEM.078	Physical Object	hasConcreteEntityType	PHYSICAL_OBJECT
R.LUO.SEM.079	Organic Object	hasConcreteEntityType	ORGANIC_OBJECT
R.LUO.SEM.080	Living Entity	hasConcreteEntityType	LIVING_ENTITY
R.LUO.SEM.081	Substance	hasConcreteEntityType	SUBSTANCE
R.LUO.SEM.082	3D Location	hasLocationType	3D
R.LUO.SEM.083	Geopolitical Location	hasLocationType	GEOPOLITICAL
R.LUO.SEM.084	Area	hasLocationType	AREA
R.LUO.SEM.085	Opening	hasLocationType	OPENING
R.LUO.SEM.086	Artifactual Location	hasLocationType	ARTIFACTUAL_ LOCATION
R.LUO.SEM.087	Other Location	hasLocationType	OTHER

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SEM.088	Non-Artifactual Material	IsArtifactual	FALSE
R.LUO.SEM.089	Artifactual Material	IsArtifactual	TRUE
R.LUO.SEM.090		hasArtifactualType	ARTIFACTUAL_MATERIAL
R.LUO.SEM.091	Furniture	hasArtifactualType	FURNITURE
R.LUO.SEM.092	Clothing	hasArtifactualType	CLOTHING
R.LUO.SEM.093	Container	hasArtifactualType	CONTAINER
R.LUO.SEM.094	Artwork	hasArtifactualType	ARTWORK
R.LUO.SEM.095	Instrument	hasArtifactualType	INSTRUMENT
R.LUO.SEM.096	Money	hasArtifactualType	MONEY
R.LUO.SEM.097	Vehicle	hasArtifactualType	VEHICLE
R.LUO.SEM.098	Semiotic Artifact	hasArtifactualType	SEMIOTIC_ARTIFACT
R.LUO.SEM.099	Other Artifact	hasArtifactualType	OTHER
R.LUO.SEM.100	Artifactual Food	hasArtifactualType	ARTIFACTUAL_FOOD
R.LUO.SEM.101		hasFoodType	ARTIFACTUAL
R.LUO.SEM.102	Non-Artifactual Food	hasFoodType	NON-ARTIFACTUAL
R.LUO.SEM.103	Flavouring	hasFoodType	FLAVOURING
R.LUO.SEM.104	Animal	hasLivingEntityType	ANIMAL
R.LUO.SEM.105	Human	hasLivingEntityType	HUMAN
R.LUO.SEM.106	Vegetal Entity	hasLivingEntityType	VEGETAL
R.LUO.SEM.107	Micro-Organism	hasLivingEntityType	MICRO
R.LUO.SEM.108	Earth Animal	hasAnimalType	EARTH
R.LUO.SEM.109	Air Animal	hasAnimalType	AIR
R.LUO.SEM.110	Water Animal	hasAnimalType	WATER
R.LUO.SEM.111	People	hasHumanType	PEOPLE
R.LUO.SEM.112	Role	hasHumanType	ROLE
R.LUO.SEM.113	Agent Of Activity	hasHumanType	AGENT
R.LUO.SEM.114	Professional	hasHumanType	PROFESSIONAL
R.LUO.SEM.115	Ideo	hasRoleType	IDEO
R.LUO.SEM.116	Kinship	hasRoleType	KINSHIP
R.LUO.SEM.117	Social Status	hasRoleType	SOCIAL_STATUS
R.LUO.SEM.118	Agent Of Temporary Activity	performsPersistentActivity	FALSE
R.LUO.SEM.119	Agent Of Persistent Activity	performsPersistentActivity	TRUE
R.LUO.SEM.120	Plant	hasVegetalType	PLANT
R.LUO.SEM.121	Flower	hasVegetalType	FLOWER

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SEM.122	Fruit	hasVegetalType	FRUIT
R.LUO.SEM.123	Substance	hasSubstanceType	NATURAL
R.LUO.SEM.124	Natural Substance	hasSubstanceType	FOOD
R.LUO.SEM.125	Food Substance	hasSubstanceType	DRINK
R.LUO.SEM.126	Artifactual Drink	isArtifactual	TRUE
R.LUO.SEM.127	Domain	hasAbstractEntityType	DOMAIN
R.LUO.SEM.128	Time	hasAbstractEntityType	TIME
R.LUO.SEM.129	Moral Standard	hasAbstractEntityType	MORAL_STANDARD
R.LUO.SEM.130	Cognitive Fact	hasAbstractEntityType	COGNITIVE_FACT
R.LUO.SEM.131	Movement Of Thought	hasAbstractEntityType	MOVEMENT_OF_THOUGHT
R.LUO.SEM.132	Institution	hasAbstractEntityType	INSTITUTION
R.LUO.SEM.133	Convention	hasAbstractEntityType	CONVENTION
R.LUO.SEM.134	General Domain	hasDomainType	GENERAL
R.LUO.SEM.135	Food Domain	hasDomainType	FOOD
R.LUO.SEM.136	Agriculture-Fishing-Forestry	hasDomainType	AGRIC-FISH-FOREST
R.LUO.SEM.137	Business Domain	hasDomainType	BUSINESS
R.LUO.SEM.138	Industry Domain	hasDomainType	INDUSTRY
R.LUO.SEM.139	Construction Domain	hasDomainType	CONSTRUCTION
R.LUO.SEM.140	Politics-Government	hasDomainType	POLITICS
R.LUO.SEM.141	Sciences	hasDomainType	SCIENCES
R.LUO.SEM.142	Health-Medicine	hasDomainType	HEALTH
R.LUO.SEM.143	Military Domain	hasDomainType	MILITARY
R.LUO.SEM.144	Home-Garden	hasDomainType	HOME
R.LUO.SEM.145	Education Domain	hasDomainType	EDUCATION
R.LUO.SEM.146	Sports-Leisure	hasDomainType	SPORTS
R.LUO.SEM.147	Arts	hasDomainType	ARTS
R.LUO.SEM.148	Religion Domain	hasDomainType	RELIGION
R.LUO.SEM.149	Transport Domain	hasDomainType	TRANSPORT
R.LUO.SEM.150	Law Domain	hasDomainType	LAW
R.LUO.SEM.151	Themes	hasDomainType	THEMES
R.LUO.SEM.152	Service Industry Domain	hasIndustryDomain	SERVICE
R.LUO.SEM.153	Craft Industry Domain	hasIndustryDomain	CRAFT
R.LUO.SEM.154	Manufacturing Industry Domain	hasIndustryDomain	MANUFACTURING
R.LUO.SEM.155	Material Process	hasActionType	MATERIAL

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LUO.SEM.156	Mental Process	hasActionType	MENTAL
R.LUO.SEM.157	Relational Process	hasActionType	RELATIONAL
R.LUO.SEM.158	Other Process	hasActionType	OTHER
R.LUO.SEM.159	Dispositive Material Process	hasMaterialProcessType	DISPOSITIVE
R.LUO.SEM.160	Creative Material Process	hasMaterialProcessType	CREATIVE
R.LUO.SEM.161	Perception Process	hasMentalProcessType	PERCEPTION
R.LUO.SEM.162	Cognition Process	hasMentalProcessType	COGNITION
R.LUO.SEM.163	Affection Process	hasMentalProcessType	AFFECTION
R.LUO.SEM.164	Intensive Process	hasRelationalProcessType	INTENSIVE
R.LUO.SEM.165	Circumstantial Process	hasRelationalProcessType	CIRCUMSTANTIAL
R.LUO.SEM.166	Possessive Process	hasRelationalProcessType	POSSESSIVE
R.LUO.SEM.167	Attributive Intensive Process	isAttributiveProcess	TRUE
R.LUO.SEM.168	Identifying Intensive Process	isAttributiveProcess	FALSE
R.LUO.SEM.169	Attributive Circumstantial Process	isAttributiveProcess	TRUE
R.LUO.SEM.170	Identifying Circumstantial Process	isAttributiveProcess	FALSE
R.LUO.SEM.171	Attributive Possessive Process	isAttributiveProcess	TRUE
R.LUO.SEM.172	Identifying Possessive Process	isAttributiveProcess	FALSE
R.LUO.SEM.173	Existential Process	hasOtherProcessType	EXISTENTIAL
R.LUO.SEM.174	Behavioural Process	hasOtherProcessType	BEHAVIOURAL
R.LUO.SEM.175	Verbal Process	hasOtherProcessType	VERBAL

Thus far, all the concepts of the LUO, which formalise the different types of linguistic units contemplated within OntoTag, have already been presented. Also their associated attributes, *ad hoc* relations, rules and axioms have been presented so far. To conclude the presentation of the LUO, some statistics about the terms that it includes are shown in the next subsection.

4.1.3.4 THE LUO STATISTICS

To conclude this section, the Linguistic Unit Ontology statistics have been summarised in Table 63.

Table 63: The LUO statistics

LUO	CONCEPTS	INSTANCES	ATTRIBUTES	RELATIONS			RULES	AXIOMS	TOTAL
				TAX.	MERON.	AD HOC			
TOP-LEVEL	6	0	2	5	0	0	0	0	13
(MORPHO) SYNTACTIC LEVEL	129	21	159	99	4	11	159	0	582
SEMANTIC LEVEL	175	0	177	124	0	28	177	0	681
TOTAL	310	21	338	228	4	39	336	0	1276
				271					

Hence, the Linguistic Unit Ontology (LUO) and its different components (or terms) have been described in the last six subsections. As commented above, this ontology contains the formalisation of the linguistic units contemplated in the model OntoTag. The formalisation of the different linguistic attributes (properties) that characterise these units is shown in the next subsection, where the Linguistic Attribute Ontology (LAO) is described as well.

4.1.4. THE LINGUISTIC ATTRIBUTE ONTOLOGY (LAO)

When analysing the different linguistic attributes included in EAGLES (1996a, 1996b) for their formalisation within OntoTag's ontologies, the first (and rather immediate) conclusions that were extracted could be stated as follows: (1) some linguistic attributes can be ascribed to more than one linguistic unit as, for instance, the attribute `Person`, which can be ascribed to the units `Verb`, `Pronoun`, and `Verbal Phrase`, amongst others; and (2) some linguistic attributes appear replicated within a given unit, but showing a rather different sense of application. This is the case of the attributes `Gender` and `Number`, which are replicated in the `Possessive Pronoun` and `Possessive Determiner` units. In effect, in these units, `Gender` is expressed as a `Grammatical Gender` or as a `Possessor Gender`¹¹⁰, and `Number` is expressed as a `Grammatical Number` or as a `Possessor Number`¹¹¹.

Therefore, the formalisation of linguistic attributes had to allow to (1) group them in order to (1.a) associate each Linguistic Unit with its corresponding Linguistic Attribute group and

¹¹⁰ Whereas the attribute `Possessive Gender` helps differentiate (i) between 'suyo' and 'suya' in Spanish, (ii) between 'sein' (Nominative, Masculine, Singular) and 'seine' (Nominative, Feminine, Singular) in German, or (iii) between 'son' and 'sa' in French, for example (this distinction does not exist in English), the attribute `Possessor Gender` helps differentiate (i) between 'his' and 'her' in English, or (ii) between 'sein' (Nominative, Masculine, Singular) and 'ihr' (Nominative, Feminine, Singular) in German, for instance.

¹¹¹ Whilst the attribute `Possessive Number` helps differentiate (i) between 'suyo' and 'suyos' in Spanish, (ii) between 'sein' (Nominative, Masculine, Singular) and 'seine' (Nominative, Plural) in German, or (iii) between 'son' and 'ses' in French, for example (this distinction does not exist in English), the attribute `Possessor Number` helps differentiate between (i) 'my' and 'our' in English, or (ii) between 'mein' (Nominative, Masculine, Singular) and 'unser' (Nominative, Masculine, Plural) in German, for instance.

(1.b) avoid an unnecessary replication of attributes (and, thus, redundancy); (2) create several replicas of a given `Linguistic Attribute` that could be ascribed to a particular unit when necessary; and (3) treat each of these replicas in a similar way, that is, assigning them the same set of possible linguistic values and ascribing them the same properties.

The aforementioned requirements for the formalisation of linguistic attributes, eventually, lead to (i) detach linguistic attributes from linguistic units; (ii) develop a dedicated ontology of linguistic attributes, namely the Linguistic Attribute Ontology (the LAO); (iii) associate the top-level concepts representing all of them in OntoTag's ontologies by means of a suitable *ad hoc* relation in the OIO (see Subsection 4.1.2.3); and (iv) link each concrete `Linguistic Attribute` to its corresponding linguistic units by means of a suitable axiom of the LAO.

This permitted not only to fulfil the three requirements stated above but also to arrange OntoTag's linguistic attributes into a taxonomy, where their properties could be conveniently inherited or inferred. More specifically, on the one hand, the first requirement was attained by formalising each `Linguistic Attribute` group by means of a different class in the LAO. On the other hand, the second and the third requirement were fulfilled by creating as many *instances* of the attributes involved as replicas were needed. Additionally, this separation between linguistic units and linguistic attributes made OntoTag's design more modular.

As for the description of the resulting concepts, relationships and other constituents of the LAO, it has been distributed into five different subsections. The first one (Subsection 4.1.4.1) is dedicated to the description of the *concepts* in the LAO and the *taxonomical relations* holding between them, as well as the *instances* of these concepts already included in this ontology; the second one (Subsection 4.1.4.2) is devoted to the *attributes* that characterise the different concepts in the ontology; the third one (Subsection 4.1.4.3) deals with the particular *ad hoc relations* that link these concepts; and the fourth one (Subsection 4.1.4.4) concerns the *rules* and *axioms* that hold for the terms in the ontology; finally, the fifth one (Subsection 4.1.4.5) presents some statistics about the constituents of this ontology.

4.1.4.1 THE LAO CONCEPTS, TAXONOMY AND INSTANCES

The different concepts of this ontology and the hierarchical relationships holding between them are described in five different subsections. The first one deals with the top-level concepts in the taxonomy, which are somehow common to all the linguistic levels contemplated in OntoTag. Each of the other four subsections has been devoted to one of these linguistic levels.

4.1.4.1.1 Top-Level Concepts and Taxonomy in the LAO

The top-level concept of the LAO, `Linguistic Attribute`, and the main concepts into which it can be subclassified are presented in this subsection. All of them, namely `Morphological Attribute`, `Syntactic Attribute`, `Semantic Attribute`, `Discourse Attribute` and `Pragmatic Attribute`¹¹², have been included in Table 64. Except for the `Morphological Attribute` concept, each of these subclassifying concepts, in turn, is the top-level concept of one of the different parts that can be distinguished within the LAO.

As for the `Morphological Attribute` concept, on the one hand, the `Morphosyntactic Attribute` concept (presented in Subsection 4.1.4.1.2) is a *Subclass-Of* `Morphological Attribute` in the LAO. On the other hand, there might be some other important morphological attributes, such as those that can be ascribed to `Affix`, `Root` and `Stem` and other morphological units. Neither these morphological units nor their associated attributes were considered in the elaboration of the EAGLES (1996a; 1996b) recommendations being formalised here. Nevertheless, they might be eventually included in the ISO standards currently under development, such as ISO/MAF (2008). Hence, in the end, this ontology might have to be extended to include all these attributes. This justifies the introduction of the `Morphological Attribute` concept into the LAO.

However, most of the morphosyntactic attributes considered in EAGLES (1996a) are ascribed also to the syntactic units included in EAGLES (1996b). Therefore, they can be considered also a type of `Syntactic Attribute`. Accordingly, as for the present release of OntoTag's ontologies, morphosyntactic attributes have been included as *subclasses of* `Syntactic Attribute` in the LAO. Yet, also the *Subclass-Of* relation that links `Morphosyntactic Attribute` to `Morphological Attribute` has been formalised within the LAO, as explained above. This entails that morphosyntactic attributes belong to both the `Morphological` and the `Syntactic Level`, which prevents these levels from being treated (completely) independently. **This**, in turn, (as with morphosyntactic units) **contributes to the refutation of one of the hypothesis underlying the development of OntoTag, namely that levels can be separated and treated independently.**

¹¹² `Morphological Attribute`, `Discourse Attribute` and `Pragmatic Attribute` have been added mainly for completeness sake and in order to enable further expansions of this ontology that fall out of the scope of the present work.

Table 64: The main concepts in the LAO and the taxonomical relations holding between them

Linguistic Attribute	Morphological Attribute
	Syntactic Attribute
	Semantic Attribute
	Discourse Attribute
	Pragmatic Attribute

The next subsections introduce the rest of the concepts in the LAO and their corresponding taxonomical relations. They are presented according to the levels to which they belong.

4.1.4.1.2 Syntactic Concepts, Taxonomy and Instances in the LAO

The concepts of the LAO that formalise the linguistic attributes of syntactic units (that is, syntactic attributes) are presented in this subsection. The taxonomical relations that hold between these concepts and also between them and the LAO top-level concepts are presented in this subsection as well. It is all summarised in Table 65.

Table 65: The syntactic concepts in the LAO and the taxonomical relations holding between them

SYNTACTIC LAO CONCEPTS				
Syntactic Attribute	Morphosyntactic Attribute	Conjugational Attribute	Other Conjugational Attribute	
		Declensional Attribute	Agreement Attribute	Person
				Politeness
				Gender
				Number
	Other Morphosyntactic Attribute	Other Declensional Attribute		
Other Syntactic Attribute				

The instances identified for these concepts (and added to the LAO) are introduced afterwards. They are included in Table 66.

Table 66: The instances of syntactic attributes in the LAO

SYNTACTIC LAO CONCEPTS	CONCEPT INSTANCES
Other Conjugational Attribute	hasAspect
	usesAuxiliary
	hasFiniteness
	hasTense
	hasMood
	hasVoice
	isReflexive
Person	hasPerson
Politeness	hasPolitenessMarkedness
Gender	hasGrammaticalGender
	hasPossessorGender
Number	hasGrammaticalNumber
	hasPossessorNumber
Other Declensional Attribute	hasCase
Other Morphosyntactic Attribute	isDefinite
	hasDegree
	hasInflectionType
	isSyntUsedAs ¹¹³
	isSeparable
	isStrong
Other Syntactic Attribute	hasSyntacticDependency
	hasActualMorphoSyntacticFunction
	hasPhraseFunction
	hasSyntacticFunction
	hasLexicalFunction
	hasSurfaceSense ¹¹⁴
	hasSurfacePolarity
	hasInterrogativeExtension
hasDeepSense ¹¹⁵	

¹¹³ This attribute characterises and subclassifies adpositions as prepositions, postpositions or circumpositions.

¹¹⁴ Also referred to as `hasGrammaticalMood` or `hasSurfaceFunction` (the suitable *Synonyms* for this class have been added to the ontology) – see Lázaro-Carreter & Tusón (1990).

These concepts and their instances have been extracted mainly from the EAGLES (1996a; 1996b) recommendations for the (morpho)syntactic annotation of corpora. However, some *subclasses* and *instances of Other Syntactic Attribute*, not (explicitly) included in these recommendations, have been extracted from other well-known sources. In particular, the `hasLexicalFunction` instance has been extracted from Melčuk (1996) and Barrios-Rodríguez (2008), and `hasSurfaceSense`, `hasSurfacePolarity`, `hasInterrogativeExtension` and `hasDeepSense` have been extracted from Lyons (1977) and Lázaro-Carreter & Tusón (1990).

Besides, some of the instances identified for *Other Syntactic Attribute* require a detailed explanation. Firstly, the `hasSyntacticDependency` instance complements the information provided by the `DependsSyntacticallyOn(Token, Token)` *ad hoc* relation defined in the (morpho)syntactic module of the LUO (Subsection 4.1.3.2). The formalisation of this attribute instance, its associated values and its complementing *ad hoc* relation enable a convenient implementation of the Syntactic Dependency Relation Labelling Layer included in EAGLES (1996b).

Secondly, the `hasActualMorphoSyntacticFunction` instance was derived from the annotations obtained from Connexor's FDG Parser. This linguistic annotation tool differentiates the theoretical morphosyntactic category associated to a *Syntactic Unit* from the actual morphosyntactic function that this *Syntactic Unit* performs in a given context. For example, the word 'robados' included in the Spanish Sentence 'Los artículos robados fueron recuperados tres días más tarde, tras la captura de los ladrones' (\approx 'The stolen properties were recovered three days later, after the capture of the thieves') is a *Verb*, but it functions as an *Adjective* in this case¹¹⁶. It must be used for annotation in conjunction with the `HasMorphoSyntacticFunctionIn(Syntactic Unit, Syntactic Unit)` *ad hoc* relation, defined as well in the syntactic module of the LUO.

Thirdly, the `hasPhraseFunction` instance is the extension of the EAGLES (1996a) attribute 'NP Function' to any kind of *Syntactic Unit* (not just a *Noun Phrase*). As with the previous instance, it must be used for annotation in conjunction with the `HasPhaseFunctionIn(Syntactic Unit, Phrase)` *ad hoc* syntactic relation.

Fourthly, the `hasSyntacticFunction` instance enables the implementation of the *syntactic function labelling layer* included in EAGLES (1996b). Also this instance must be used for annotation

¹¹⁵ Also referred to as `hasDeepFunction` (a suitable *Synonym* for this class has been added into the ontology) – see Lázaro-Carreter & Tusón (1990).

¹¹⁶ These participle verbs functioning as adjectives are the result of the abbreviation of a *Relative Clause*.

in conjunction with another *ad hoc* syntactic relation, namely `HasSyntacticFunctionIn(Syntactic Unit, Syntactic Unit)`.

Fifthly, the `hasLexicalFunction` instance complements the information provided by the `CollocatesWith(Token, Token)` *ad hoc* relation, also defined in the syntactic module of the LUO. The formalisation of these last four Other Syntactic Attribute instances, namely `hasLexicalFunction`, `hasSyntacticFunction`, `hasPhraseFunction` and `hasActualMorpho-SyntacticFunction`, together with their complementing *ad hoc* relations (and their associated values), enable a convenient implementation of the *syntactic function labelling layer* included in EAGLES (1996b)¹¹⁷.

Finally, the meaning and the use of all the Other Syntactic Attribute instances can be better understood in conjunction with the values that they can take, shown in Subsection 4.1.5.1.2.

As for the taxonomical relations that bear between these syntactic concepts, briefly, `Morphosyntactic Attribute` and `Other Syntactic Attribute` constitute a *Partition* of `Syntactic Attribute`; `Conjugational Attribute`, `Declensional Attribute` and `Other Morphosyntactic Attribute` constitute an *Exhaustive-Decomposition* of `Morphosyntactic Attribute`¹¹⁸; `Agreement Attribute` and `Other Conjugational Attribute` constitute a *Partition* of `Conjugational Attribute`; `Agreement Attribute` and `Other Declensional Attribute` constitute a *Partition* of `Declensional Attribute`; and, finally, `Person`, `Politeness`, `Gender` and `Number` constitute a *Disjoint-Decomposition* of `Agreement Attribute`.

4.1.4.1.3 Semantic Concepts, Taxonomy and Instances in the LAO

The concepts of the LAO that formalise the linguistic attributes of semantic units (that is, semantic attributes), as well as the taxonomical relations that hold (a) between these concepts and also (b) between them and the LAO top-level concepts, are presented in Table 67. As shown in this table, the inclusion of the subclass `Other Semantic Attribute` allows for claiming that an *Exhaustive-Decomposition* of `Semantic Attribute` has been achieved.

¹¹⁷ The remaining syntactic attributes, not related to the syntactic dependency relation labelling layer or the syntactic function labelling layer, help implement the syntactic feature annotation layer of EAGLES (1996b).

¹¹⁸ It cannot be a *Disjoint-Decomposition*, since `Conjugational Attribute` and `Declensional Attribute` share the instances of `Agreement Attribute`.

Table 67: The semantic concepts in the LAO and the taxonomical relations holding between them

SEMANTIC LAO CONCEPTS	
Semantic Attribute	Propositional Component Attribute
	State Of Affairs Attribute ¹¹⁹
	Quality Attribute
	Other Semantic Attribute

The instances identified for the concepts shown in Table 67 (and added to the LAO) have been included in Table 68.

Table 68: The instances of semantic attributes in the LAO

SEMANTIC LAO CONCEPTS	CONCEPT INSTANCES
Propositional Component Attribute ¹²⁰	isInstanced
	hasParticipantType
	hasSemanticRole
State Of Affairs Attribute	isDynamic
	isTelic
	isMomentaneous
	hasController
	isExperiential
Quality Attribute	isAttributive
	isPredicative

First, as far as the *instances of* Propositional Component Attribute are concerned, on the one hand, `isInstanced` captures the semantic difference between ‘(a) park’ and ‘McArthur’s Park’, or between ‘one day’ and ‘today’, for example. It fixes the Sense Tagging Layer in which the corresponding Propositional Component must be annotated, namely the Instance Semantic Annotation Layer (when `isInstanced = TRUE`) or the Concept Semantic Annotation Layer (in other case). On the other hand, `hasParticipantType` and `hasSemanticRole` detail the semantic function performed by the particular Propositional Component to which they are ascribed in a particular Proposition (i.e., Sentence). The former (`hasParticipantType`) has been derived from Halliday’s (1994; 1996) Functional Grammar and its values specify the role of the Propositional Component with respect to the

¹¹⁹ Also referred to as Action Attribute.

¹²⁰ This subclass and its instances have been extracted from Halliday’s (1994; 1996) Functional Grammar. It complements the information about the Semantic Role of a Propositional Component within its corresponding Proposition.

particular `Action` in which it participates. The latter has been extracted from Gildea & Jurafsky (2002) and its values detail the semantic roles used within the FrameNet project¹²¹. It specifies a more general semantic role of the `Propositional Component` with respect to its `Proposition`. Therefore, these two instances are complementary and both had to be formalised within the LAO (*id.* the LVO).

Second, the *instances of State Of Affairs Attribute* have been extracted from Dik (1989). They give a complementary characterisation of the LUO `Action` units to the one presented in Halliday (1994; 1996), which was used to subclassify the concept `Action`. Besides, it helps the OntoTag model become independent of Halliday's assumptions. That is why these instances were included in the LAO.

Third, the *instances of Quality Attribute* have been derived from Lázaro-Carreter & Tusón (1990). They are used, for example, in Connexor's FDG Parser and Machine Syntax to characterise both adjectives and adjectivals. Therefore, including them in the LAO was required in order to process conveniently these annotations by means of OntoTag(ger).

Finally, the class `Other Semantic Attribute` has been included in this classification for completeness sake and in order to make it easier to extend the ontology with other semantic attributes in the future (should they be found).

Up to this point, all the *concepts* of the LAO and their corresponding *taxonomical relations* have already been presented, as well as their associated *instances*. There comes the time, hence, to show the *attributes* that characterise them, the *ad hoc* relations holding between them, and the *rules* and *axioms* that further constrain their signification and application. This will be done in the next subsections.

4.1.4.2 THE LAO ATTRIBUTES

The attributes associated to the concepts of the LAO are presented in Table 69. All of them are class attributes. They have been grouped according to their level for their presentation in this table, which shows, for each attribute, the level and the concept to which it belongs and the type of values that it can take. Some of the attributes associated to the concepts in this table are associated to its subclasses as well. In these subclasses, these attributes take a value that is particular to that subclass. Therefore, it is defined by means of a corresponding rule in Subsection 4.1.4.4. These subclass-attribute associations have not been included here in order to avoid redundancy and for the sake of conciseness.

¹²¹ See <http://framenet.icsi.berkeley.edu/> for details.

Table 69: Attributes associated to the concepts within the LAO

LEVEL	CONCEPT	ATTRIBUTE	VALUE TYPE
Top Level	Linguistic Attribute	LinguisticAttType	{MORPHOLOGICAL, SYNTACTIC, SEMANTIC, DISCOURSE-RELATED, PRAGMATIC}
Syntactic Level	Syntactic Attribute	isMSAttribute	BOOLEAN
	Morphosyntactic Attribute	hasMSAttType	{CONJUGATIONAL, DECLENSIONAL, OTHER}
	Conjugational Attribute	isAgreementAttribute	BOOLEAN
	Declensional Attribute	isAgreementAttribute	BOOLEAN
	Agreement Attribute	hasAgreementAttType	{PERSON_TYPE, POLITENESS_TYPE, GENDER_TYPE, NUMBER_TYPE}
Semantic Level	Semantic Attribute	SemanticAttType	{PROP-COMP, SOA, QUALITY, OTHER}

4.1.4.3 THE LAO *AD HOC* RELATIONS

There are no *ad hoc* relations holding between the concepts of this ontology. There are, nevertheless, some other *ad hoc* relations holding between the concepts of this ontology and other ontologies of OntoTag. These other *ad hoc* relations have been either (i) represented in the OIO (see Section 4.1.2.5.1) or (ii) formalised by means of axioms and rules (see Subsection 4.1.4.4). Therefore, no further comment is required on this issue within the LAO.

4.1.4.4 THE LAO RULES AND AXIOMS

After describing in detail all the concepts of the ontology, their associated taxonomies (and/or meronomies), their attributes, and their interrelations, the formal *rules* and *axioms* that hold within the ontology must be defined as well. These formal *rules* and *axioms* are used for constraint checking and for inferring attribute values. Hence, they are a key component in heavyweight ontologies.

As far as the *rules* of the LAO are concerned, they all determine the values that the class attributes defined in the previous subsections take within a given concept. These values are presented, using the tabular notation introduced in Subsection 4.1.3.2.4, in Table 70.

Table 70: Rules associated to the attribute values of the concepts within the LAO

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LAO.MORPH.001	Morphological Attribute	LinguisticAttType	MORPHOLOGICAL

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LAO.SYN.001	Syntactic Attribute	LinguisticAttType	SYNTACTIC
R.LAO.SYN.002	Morphosyntactic Attribute	isMSAttribute	TRUE
R.LAO.SYN.003	Other Syntactic Attribute	isMSAttribute	FALSE
R.LAO.SYN.004	Conjugational Attribute	hasMSAttType	CONJUGATIONAL
R.LAO.SYN.005	Declensional Attribute	hasMSAttType	DECLENSIONAL
R.LAO.SYN.006	Other Morphosyntactic Attribute	hasMSAttType	OTHER
R.LAO.SYN.007	Other Conjugational Attribute	isAgreementAttribute	FALSE
R.LAO.SYN.008	Agreement Attribute	isAgreementAttribute	TRUE
R.LAO.SYN.009	Other Declensional Attribute	isAgreementAttribute	FALSE
R.LAO.SYN.010	Person	hasAgreementAttType	PERSON_TYPE
R.LAO.SYN.011	Politeness	hasAgreementAttType	POLITENESS_TYPE
R.LAO.SYN.012	Gender	OtherAgreementAttType	GENDER_TYPE
R.LAO.SYN.013	Number	OtherAgreementAttType	NUMBER_TYPE
R.LAO.SEM.001	Semantic Attribute	LinguisticAttType	SEMANTIC
R.LAO.SEM.002	Propositional Component Attribute	SemanticAttType	PROP-COMP
R.LAO.SEM.003	State Of Affairs Attribute	SemanticAttType	SOA
R.LAO.SEM.004	Quality Attribute	SemanticAttType	QUALITY
R.LAO.SEM.005	Other Semantic Attribute	SemanticAttType	OTHER

As for the axioms of the LAO, they have been included in Table 71. In this table, the corresponding axioms have been specified following a simplified notation, for the sake of space, expressiveness and clarity. All of them are expressed according to the usual formalism used in Mathematics and Logic for representing formulae. All these axioms formalise the relationships between linguistic units and their corresponding attributes. That is, they constrain the application of the attributes in the LAO to those units in the LUO for which they are defined.

Table 71: The LAO axioms

AXIOM IDENTIFIER	AXIOM
A.LAO.SYN.01	$\forall x ((Noun(x) \vee Noun_Phrase(x) \vee Nominal_Clause(x) \vee Adjective(x) \vee Adjective_Phrase(x) \vee Comparative_Clause(x) \vee Relative_Clause(x) \vee Pronoun-Determiner(x) \vee Article(x) \vee Numeral(x)) \leftrightarrow is_Declensional(x))$
A.LAO.SYN.02	$\forall x ((Verb(x) \vee Verbal_Multiword_Token(x) \vee Verb_Phrase(x)) \leftrightarrow is_Conjugational(x))$
A.LAO.SYN.03	$\forall x, y ((is_Conjugational(x) \wedge Conjugational_Attribute(y)) \rightarrow Has(x, y))$

A.LAO.SYN.04	$\forall x ((is_Declensional(x) \vee is_Conjugational(x) \vee Residual(x)) \rightarrow (Has(x, hasGrammaticalGender) \wedge Has(x, hasGrammaticalNumber)))$
A.LAO.SYN.05	$\forall x (Possessive_Pronoun-Determiner(x) \rightarrow (Has(x, hasPossessorGender) \wedge Has(x, hasPossessorNumber)))$
A.LAO.SYN.06	$\forall x ((Pronoun-Determiner(x) \vee is_Conjugational(x)) \rightarrow Has(x, hasPerson))$
A.LAO.SYN.07	$\forall x (Pronoun-Determiner(x) \rightarrow Has(x, hasPolitenessMarkedness))$
A.LAO.SYN.08	$\forall x, y ((is_Declensional(x) \wedge Other_Declensional_Attribute(y)) \rightarrow Has(x, y))$
A.LAO.SYN.09	$\forall x (Noun(x) \rightarrow Has(x, isDefinite))$
A.LAO.SYN.10	$\forall x ((Adjective(x) \vee Adverb(x)) \rightarrow Has(x, hasDegree))$
A.LAO.SYN.11	$\forall x (Adjective(x) \rightarrow Has(x, hasInflectionType))$
A.LAO.SYN.12	$\forall x (Adposition(x) \rightarrow Has(x, isSyntUsedAs))$
A.LAO.SYN.13	$\forall x (Verb(x) \rightarrow Has(x, isSeparable))$
A.LAO.SYN.14	$\forall x (Pronoun(x) \rightarrow Has(x, isStrong))$
A.LAO.SYN.15	$\forall x (Token(x) \rightarrow (Has(x, hasSyntacticDependency) \wedge Has(x, hasLexicalFunction)))$
A.LAO.SYN.16	$\forall x (Syntactic_Unit(x) \rightarrow (Has(x, hasActualMorphoSyntacticFunction) \wedge Has(x, hasPhraseFunction) \wedge Has(x, hasSyntacticFunction)))$
A.LAO.SYN.17	$\forall x (Clause(x) \vee Sentence(x) \rightarrow Has(x, hasSurfaceSense) \wedge Has(x, hasSurfacePolarity) \wedge Has(x, hasInterrogativeExtension) \wedge Has(x, hasDeepSense))$
A.LAO.SEM.01	$\forall x, y ((Semantic_Unit(x) \wedge Semantic_Unit_Attribute(y)) \rightarrow Has(x, y))$
A.LAO.SEM.02	$\forall x, y ((Propositional_Component(x) \wedge Propositional_Component_Attribute(y)) \rightarrow Has(x, y))$
A.LAO.SEM.03	$\forall x, y ((State_Of_Affairs(x) \wedge State_Of_Affairs_Attribute(y)) \rightarrow Has(x, y))$
A.LAO.SEM.04	$\forall x, y ((Quality(x) \wedge Quality_Attribute(y)) \rightarrow Has(x, y))$

Thus far, all the concepts in the LAO have already been presented. They formalise the different types the different types of linguistic attributes that can be ascribed to the linguistic units contemplated within OntoTag. Also their associated (ontological) attributes, *ad hoc* relations, rules and axioms have been presented up to this point. To conclude the presentation of the LAO, some numeric statistics are shown in the next subsection about the components that it includes.

4.1.4.5 THE LAO STATISTICS

To conclude this section, the Linguistic Attribute Ontology statistics have been summarised in Table 72.

Table 72: The LAO statistics

LAO	CONCEPTS	INSTANCES	ATTRIBUTES	RELATIONS			RULES	AXIOMS	TOTAL
				TAX.	MERON.	AD HOC			
TOP-LEVEL	6	0	0	0	0	0	0	0	6
(MORPHO) SYNTACTIC LEVEL	12	29	14	7	0	0	14	17	93
SEMANTIC LEVEL	6	29	8	1	0	0	8	5	57
TOTAL	24	58	22	8	0	0	22	22	156
				8					

4.1.5. THE LINGUISTIC VALUE ONTOLOGY (LVO)

The separate formalisation of linguistic units and linguistic attributes (that is, in separate and dedicated ontologies) entailed also the separate formalisation of linguistic values within their own and separate ontology, for the sake of modularity. Thus, the present ontology, the Linguistic Value Ontology (LVO), contains the different linguistic values identified for the linguistic attributes included in the LAO.

This ontology was developed as follows. As with linguistic attributes, (1) linguistic values were grouped according to the attributes that could take them; (2) each of the resulting Linguistic Value groups was formalised as a different class within the LVO; (3) each different Linguistic Value was formalised as an *Instance-Of* the classes formalised in the previous step; (4) the top-level concept of the LAO (Linguistic Attribute) was associated to the top-level concept of the LVO (Linguistic Value) within the OIO by means of a suitable *ad hoc* relation (Takes(Linguistic Attribute, Linguistic Value) – see Subsection 4.1.2.5.1); and (5) each concrete Linguistic Value was linked to its corresponding Linguistic Attribute(s) by means of a suitable axiom of the LVO. As a side effect, also in this case, the formalisation of the

different classes of linguistic values allowed (1) to arrange them into a taxonomy and (2) to use inheritance or inference mechanisms to determine their corresponding properties within the ontology.

As for the description of the resulting concepts, relationships and other constituents of the LVO, it has been distributed into five different subsections. The first one (Subsection 4.1.5.1) is dedicated to the description of the *concepts* in the LVO and the *taxonomical relations* holding between them, as well as the *instances* of these concepts already included in this ontology; the second one (Subsection 4.1.5.2) is devoted to the *attributes* that characterise the different concepts in the ontology; the third one (Subsection 4.1.5.3) deals with the particular *ad hoc relations* that link these concepts; and the fourth one (Subsection 4.1.5.4) concerns the *rules* and *axioms* that hold for the terms in the ontology; finally, the fifth one (Subsection 4.1.5.5) presents some statistics about the constituents of this ontology.

4.1.5.1 THE LVO CONCEPTS, TAXONOMY AND INSTANCES

The different concepts of this ontology and the hierarchical relationships holding between them are described in five different subsections. The first one deals with the top-level concepts in the taxonomy, which are somehow common to all the linguistic levels contemplated in OntoTag. Each of the other four subsections has been devoted to one of these linguistic levels.

4.1.5.1.1 Top-Level Concepts and Taxonomy in the LVO

The top-level concept of the LVO, Linguistic Value, and the main concepts into which it can be subclassified are presented in this subsection. All of them, namely Morphological Value, Syntactic Value, Semantic Value, Discourse Value and Pragmatic Value, have been included in Table 73. Only the Syntactic Value and the Semantic Value concepts, in turn, are the top-level concept of one of the two different sub-ontologies that can be distinguished within the LVO. The remaining ones have been added for completeness sake to the ontology, but the development of their corresponding sub-ontologies fell out of the scope of the present work.

In addition, the Morphological Value concept is the super-class of the Morphosyntactic Value concept (presented in Subsection 4.1.5.1.2) in the LVO. Besides, there might be some other important morphological values, such as those that can be taken by the attributes ascribed to Affix, Root and Stem and other morphological units. Neither these morphological units nor their associated attributes and values were considered in the elaboration of the EAGLES (1996a; 1996b) recommendations being formalised here. Nevertheless, they might be eventually included in the ISO

standards currently under development, such as ISO/MAF (2008). Hence, in the end, this ontology might have to be extended to include all these values.

However, most of the morphosyntactic values considered in EAGLES (1996a) can be taken by some attributes that are also ascribed to the syntactic units included in EAGLES (1996b). Therefore, they can be considered also a type of *Syntactic Value*. Accordingly, as for the present release of OntoTag's ontologies, morphosyntactic values have been included as *subclasses of Syntactic Value* in the LVO. Yet, also the *Subclass-Of* relation that links *Morphosyntactic Value* to *Morphological Value* has been formalised within the LVO, as explained above.

Table 73: The main concepts in the LVO and the taxonomical relations holding between them

Linguistic Value	Morphological Value
	Syntactic Value
	Semantic Value
	Discourse Value
	Pragmatic Value

These are the top-level concepts included in the LVO. The rest of its concepts and their corresponding taxonomical relations are presented in the next subsections, according to the levels to which they belong.

4.1.5.1.2 Syntactic Concepts, Taxonomy and Instances in the LVO

The concepts of the LVO that formalise the values of syntactic attributes (that is, the syntactic values) are presented in this subsection. The taxonomical relations that hold between these concepts and also between them and the LVO top-level concepts are presented in this subsection as well. It is all summarised in Table 74. The instances identified for these concepts (and added to the LVO) are introduced afterwards. They are included in Table 75, Table 76, Table 77, Table 78, Table 79, Table 80 and Table 81. These concepts and their instances have been extracted mainly from the EAGLES (1996a; 1996b) recommendations. However, the *subclasses* and *instances of Other Syntactic Value*, not (explicitly) included in these recommendations, have been extracted from other well-known sources.

Table 74: The syntactic concepts in the LVO and the taxonomical relations holding between them

SYNTACTIC LAO CONCEPTS					
Syntactic Value	Morphosyntactic Value	Conjugational Value	Aspect Value		
			Auxiliary Value		
			Finiteness Value		
			Tense Value		
			Mood Value		
			Voice Value		
		Reflexivity Value	BOOLEAN VALUE		
		Declensional Value	Agreement Value	Morphosyntactically Encoded Markedness Value	Person Value
				Other Agreement Value	Politeness Value
			Other Declensional Value	Case Value	Gender Value
	Number Value				
	Other Morphosyntactic Value	Definiteness Value	BOOLEAN VALUE		
		Degree Value ¹²²			
		Inflection Type Value ¹²³			
		Syntactically Used As Value			
Separability Value		BOOLEAN VALUE			
Strength Value		BOOLEAN VALUE			
Other Syntactic Value	Syntactic Dependency Value				
	Actual Morphosyntactic Function Value				
	Phrase Function Value				
	Syntactic Function Value				
	Lexical Function Value				

¹²² Applied to annotate whether an Adjective presents a Positive, a Comparative or a Superlative form.

¹²³ Applied, for example in German, to annotate whether a word form presents a Weak-Inflection, or a Strong-Inflection.

	Surface Sense Value ¹²⁴
	Surface Polarity Value
	Interrogative Extension Value
	Deep Sense Value ¹²⁵

In particular, (1) the instances of the subclasses of *Morphosyntactic Value* shown in Table 75 formalise rather straightforwardly (most of) the different attribute values considered in EAGLES (1996a); (2) the *Syntactic Dependency Value* instances shown in Table 76 have been extracted from the Connexor’s FDG Parse and Machine Syntax annotation schemas¹²⁶; (3) the *Morphosyntactic Function Value* instances shown in Table 77 have been derived from the major morphosyntactic categories included in EAGLES (1996a); (4) the *Phrase Function Value* instances shown in Table 78 have been extracted from the values of the ‘NP function’ attribute of these same recommendations; (5) the *Syntactic Function Value* instances shown in Table 79 have been extracted from Downing & Locke (2002), but they can also be found throughout the literature (in Greenbaum & Quirk (1990), for example); (6) the *Lexical Function Value* instances shown in Table 80 have been extracted from Barrios-Rodríguez (2008) – they are a sample of the most salient and/or frequent lexical functions presented in Melčuk (1996); and (7) the instances of *Surface Sense Value*, *Surface Polarity Value*, *Interrogative Extension Value* and *Deep Sense Value* shown in Table 81 have been extracted from Lyons (1977) and Lázaro-Carreter & Tusón (1990).

Table 75: The instances of the subclasses of *Morphosyntactic Value* in the LVO

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
Aspect Value	IMPERFECTIVE
	PERFECTIVE
Auxiliary Value	HAVE
	BE
	NOT APPLICABLE ¹²⁷
Finiteness Value	FINITE
	NON-FINITE
	UNMARKED

¹²⁴ See Lázaro-Carreter & Tusón (1990). Also referred to as *Grammatical Mood Value* or *Surface Function Value* (the suitable *Synonyms* for this class have been added into the ontology).

¹²⁵ See Lázaro-Carreter & Tusón (1990). Also referred to as *Deep Function Value* (a suitable *Synonym* for this class has been added into the ontology).

¹²⁶ See <http://193.185.105.50/demo/machinese/doc/enfdg3-tags.html>, and also www.connexor.eu/technology/machinese/demo/syntax.

¹²⁷ Used to mark a *Verb* that needs not being accompanied by an *Auxiliary* to be considered a completely lexical verbal form (for example, the form ‘Excuse’ in ‘Excuse me’).

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
Tense Value	FUTURE
	IMPERFECT
	PAST
	PRESENT
Mood Value	CONDITIONAL
	GERUND
	IMPERATIVE
	INDICATIVE
	INFINITIVE
	-ING FORM
	PARTICIPLE
	SUBJUNCTIVE
Voice Value	SUPINE
	ACTIVE
	PASSIVE
Gender Value	MEDIO-PASSIVE
	FEMININE
	MASCULINE
	NEUTER
	COMMON
Number Value	NOT APPLICABLE ¹²⁸
	PLURAL
Person Value	SINGULAR
	FIRST
	SECOND
Politeness Markedness Value	THIRD
	UNMARKED
	POLITE
Case Value ¹²⁹	UNPOLITE
	ABLATIVE
	ACCUSATIVE
	DATIVE

¹²⁸ Used to mark a Verb that cannot be considered to have an associated Gender (such as Finite verbal forms in Spanish).

¹²⁹ Only the main case values included in EAGLES (1996a) have been included here, minimally extended to cover Slavic languages. Many more case values can be found in the ISO TC 37 Data Category Registry site (<http://www.isocat.org/>). They might be easily added to the ontology when required – they will be just new instances of this ontology.

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
	GENITIVE
	INDECLINABLE
	INSTRUMENTAL
	LOCATIVE
	NOMINATIVE
	OBLIQUE
	PREPOSITIONAL
	VOCATIVE
Degree Value	POSITIVE
	COMPARATIVE
	SUPERLATIVE
Inflection Type Value	MIXED
	STRONG-INFLECTION
	WEAK-INFLECTION
Syntactically Used As Value	PREPOSITION
	POSTPOSITION
	CIRCUMPOSITION

Table 76: The instances of Syntactic Dependency Value in the LVO

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
Syntactic Dependency Value	MAIN
	VERB CHAIN LINK
	PREPOSITIONAL MARKER
	PREVERBAL PARTICLE
	VERB PARTICLE
	SUBJECT DEPENDENCY
	OBJECT DEPENDENCY
	SUBJECT COMPLEMENT DEPENDENCY
	INDIRECT OBJECT DEPENDENCY
	OBJECT COMPLEMENT DEPENDENCY
	COPREDICATIVE
	VOCATIVE
	TIME DEPENDENCY
	DURATION DEPENDENCY

	FREQUENCY DEPENDENCY
	QUANTITY DEPENDENCY
	MANNER DEPENDENCY
	LOCATION DEPENDENCY
	SOURCE DEPENDENCY
	GOAL DEPENDENCY
	CONTINGENCY DEPENDENCY ¹³⁰
	REASON DEPENDENCY
	PURPOSE DEPENDENCY
	CONDITION DEPENDENCY
	COMITATIVE
	CLAUSE ADVERBIAL
	CLAUSE INITIAL ADVERBIAL
	ATTRIBUTIVE ADVERBIAL
	ATTRIBUTIVE NOMINAL
	QUANTIFIER DEPENDENCY
	DETERMINER DEPENDENCY
	ADJECTIVAL POSTMODIFIER
	OTHER POSTMODIFIER
	NEGATOR
	PP ATTACHMENT

Table 77: The instances of Morphosyntactic Function value in the LVO

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
Morphosyntactic Function Value	ATTRIBUTIVE FUNCTION (ADJECTIVAL FUNCTION)
	ADPOSITIVE FUNCTION
	ADVERBIAL FUNCTION
	CONJUNCTIVE FUNCTION
	DETERMINATIVE FUNCTION
	NOMINAL FUNCTION
	PREDICATIVE FUNCTION
	PRONOMINAL FUNCTION
	QUANTIFYING FUNCTION
	INDETERMINATE FUNCTION

¹³⁰ This is an underspecification of both a Purpose Dependency and a Reason Dependency.

Table 78: The instances of Phrase Function Value in the LVO

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
Phrase Function Value	HEAD
	MODIFIER ¹³¹
	PREMODIFIER
	POSTMODIFIER

Table 79: The instances of Syntactic Function Value in the LVO

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
Syntactic Function Value	ADJUNCT
	CONJUNCT
	DIRECT OBJECT
	DISJUNCT
	INDIRECT OBJECT
	OBJECT COMPLEMENT (OBJECT PREDICATIVE)
	PREDICATE ¹³²
	SUBJECT

Table 80: The instances of Lexical Function Value in the LVO

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
Lexical Function Value	A ₀
	A _i
	Anti
	Bon
	Cap
	Caus
	Cont
	Degrad
	Fact ₀
	Fin
	Func ₀
	Func _i
	Incep
Involv	

¹³¹ This is an underspecification of both Premodifier and Postmodifier.

¹³² This instance was added for the sake of completeness, in order to annotate the usual Syntactic Function of a Verb Phrase, for example.

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
	Labor
	Labreal
	Liqu
	Magn
	Manif
	Minus
	Mult
	Oper
	Perm
	Plus
	Real _i
	S ₀
	S _i
	Sing

Table 81: The instances of the subclasses of Other Syntactic Value in the LVO

SYNTACTIC LVO CONCEPTS	CONCEPT INSTANCES
Surface Sense Value	DECLARATIVE
	INTERROGATIVE
	EXCLAMATIVE
	IMPERATIVE SURFACE SENSE ¹³³
Surface Polarity Value	AFFIRMATIVE
	NEGATIVE
Interrogative Extension Value	TOTAL (POLAR)
	PARTIAL (NON-POLAR)
Deep Sense Value	INDICATIVE DEEP SENSE
	IMPERATIVE DEEP SENSE ¹³⁴
	DUBITATIVE
	DESIDERATIVE ¹³⁵

As for the taxonomical relations that bear between these syntactic values, on the one hand, there exist several classes whose instances are Boolean values. This prevents the subclassifications of most

¹³³ To see the difference between the Surface Sense Value and the Deep Sense Value concepts, just consider the following Sentence: 'Will you close the window, please?', which has a clear INTERROGATIVE Surface Sense Value, but also an undeniable IMPERATIVE DEEP SENSE value.

¹³⁴ Also known as EXHORTATIVE (Lázaro & Tusón, 1990).

¹³⁵ Also known as OPTATIVE (Lázaro & Tusón, 1990).

of the concepts from being formalised as *Disjoint-Decompositions*. On the other hand, all these subclassifications have been formalised as *Exhaustive-Decompositions* of the corresponding parent concepts.

Up to this point, the top-level and the syntactic concepts (and taxonomy) of the LVO have already been presented. There comes the time, hence, to present the semantic concepts of the LVO and their corresponding taxonomical relations. This will be done in the next subsection.

4.1.5.1.3 Semantic Concepts, Taxonomy and Instances in the LVO

The concepts of the LVO that formalise the values of semantic attributes (that is, the semantic values) are presented in this subsection. As shown in Table 82, they have been grouped into classes and arranged according to the semantic attributes that can take them. Accordingly, the subclasses of `Semantic Value` constitute so far an *Exhaustive-Decomposition* of this concept. This has been further guaranteed by the inclusion of the `Other Semantic Value` concept in the ontology (for which no instances have been defined yet). They can be neither a *Disjoint-Decomposition* nor a *Partition* of `Semantic Value`, since the Boolean set of values is common to most of the concepts of this *Exhaustive-Decomposition*.

Table 82: The semantic concepts in the LVO and the taxonomical relations holding between them

SEMANTIC LVO CONCEPTS		
Semantic Value	State Of Affairs Value	
	Quality Value	
	Propositional Component Value	Is Instanced Value
		Participant Type Value
		Semantic Role Value
	Other Semantic Value	
		BOOLEAN VALUE

The instances identified for the `Participant Type Value` class and for the `Semantic Role Type Value` class (and added to the LVO) are introduced afterwards. They are included in Table 83. The former subclass and its instances have been extracted from Halliday's (1994; 1996) *Functional Grammar*. The latter subclass and its instances have been extracted from Gildea & Jurafsky (2002). All of them altogether are an abstraction of the semantic roles included in the FrameNet

project¹³⁶, which assumes that no canonical set of semantic roles can be determined (beforehand). As can be deduced from Table 82 and Table 83, the subclasses of Propositional Component Value constitute a *Partition* of this concept thus far.

Table 83: The instances of semantic values in the LVO

SEMANTIC LVO CONCEPTS	CONCEPT INSTANCES
Participant Type Value	ACTOR
	ATTRIBUTE
	BEHAVER
	CARRIER
	EXISTENT
	GOAL
	IDENTIFIED
	IDENTIFIER
	NULL
	PHENOMENON
	RECIPIENT
	SAYER
	SENDER
	TARGET
	TOKEN
VALUE	
Semantic Role Value (Propositional Function Value)	AGENT ROLE
	AREA ROLE
	CAUSE ROLE
	DIRECTION ROLE
	DISTANCE ROLE
	EXPERIENCER ROLE
	FORCE ROLE
	GOAL ROLE
	INSTRUMENT ROLE
	LOCATION ROLE
	MANNER ROLE
	NULL ROLE
	PATH ROLE
	PATIENT ROLE
	PERCEPT ROLE
	PROPOSITION ROLE

¹³⁶ See <http://framenet.icsi.berkeley.edu/> for details.

SEMANTIC LVO CONCEPTS	CONCEPT INSTANCES
	RESULT ROLE
	SOURCE ROLE
	STATE ROLE
	THEME ROLE
	TOPIC ROLE ¹³⁷

Up to this point, all the *concepts* of the LVO and their corresponding *taxonomical relations* have already been presented, as well as their associated *instances*. The *attributes* that characterise them, the *ad hoc* relations holding between them, and the *rules* and *axioms* that further constrain their signification and application are shown in the next subsections.

4.1.5.2 THE LVO ATTRIBUTES

The attributes associated to the concepts included in the LVO are included in Table 84. This table shows, for each attribute, the level and the concept to which it is ascribed and the values that it can take. Some of the attributes associated to the concepts in this table are associated to its subclasses as well. In these subclasses, these attributes take a value that is particular to that subclass. Therefore, it is defined by means of a corresponding rule in Section 4.1.5.4. These subclass-attribute associations have not been included here in order to avoid redundancy and for the sake of conciseness.

Table 84: Attributes of the Pragmatic Level concepts defined within the LVO

LEVEL	CONCEPT	ATTRIBUTE	VALUE TYPE
Top-Level	Linguistic Value	LinguisticValueType	{MORPHOLOGICAL, SYNTACTIC, SEMANTIC, DISCOURSE-RELATED, PRAGMATIC}
Syntactic	Syntactic Value	isMSValue	BOOLEAN
	Morphosyntactic Value	MSValueType	{CONJUGATIONAL, DECLENSIONAL, OTHER}
	Conjugational Value	isAgreementValue	BOOLEAN
	Other Conjugational Value	OtherConjValType	{ASPECT, AUXILIARY, FINITENESS, TENSE, MOOD, VOICE, REFLEXIVITY}
	Declensional Value	isAgreementValue	BOOLEAN
	Agreement Value	AgreementValueType	{PERSON, POLITENESS, GENDER, NUMBER}

¹³⁷ These are not the only existing instances of semantic roles. However, these are the most frequently used ones (see <http://framenet.icsi.berkeley.edu/>). An exhaustive list of semantic roles (as such) could not be found when developing this ontology. However, a new ISO standard proposal has been issued recently and the LVO will be updated to include its categories when this standard is eventually released.

LEVEL	CONCEPT	ATTRIBUTE	VALUE TYPE
	Other Morphosyntactic Value	OtherMSValType	{DEFINITENESS, DEGREE, INFLECTION, USED_AS, SEPARABILITY, STRENGTH}
	Other Syntactic Value	OtherSynValType	{ACTUAL_M-S_FUNCTION, PHRASE_FUNCTION, SYNTACTIC_FUNCTION, SYNTACTIC_DEPENDENCY, LEXICAL_FUNCTION, SURFACE_SENSE, DECLARATIVE, INTERROGATIVE_EXTENSION, DEEP_SENSE}
Semantic	Semantic Value	SemanticValueType	{EVENT, QUALITY, PROPOSITIONAL_COMPONENT, OTHER}
	Propositional Component Value	PropCompValueType	{INSTANCED, PARTICIPANT, SEMANTIC_ROLE}

4.1.5.3 THE LVO *Ad Hoc* RELATIONS

There are no *ad hoc* relations holding between the concepts of this ontology. There are, nevertheless, some other *ad hoc* relations holding between the concepts of this ontology and other ontologies of OntoTag. These other *ad hoc* relations have been either (i) represented in the OIO (see Section 4.1.2.5.1) or (ii) formalised by means of axioms and rules (see Subsection 4.1.5.4). Therefore, no further comment is required on this respect within the LVO.

4.1.5.4 THE LVO RULES AND AXIOMS

After describing in detail all the *concepts* of the ontology, their associated *taxonomies* (and/or *meronomies*), their *attributes* and their *ad hoc relations*, the formal *rules* and *axioms* that hold within the ontology must be defined as well. These formal *rules* and *axioms* are used for constraint checking and for inferring attribute values. Hence, as commented in the description of the LAO, they are a key component in (heavyweight) ontologies.

As far as the rules of the LVO are concerned, they all determine the values that the class attributes defined in the previous subsections take within a given concept. These values are presented, using the tabular notation introduced in Subsection 4.1.3.2.4, in Table 85.

Table 85: Rules associated to the attribute values of the concepts defined within the LVO

RULE ID	CONCEPT	ATTRIBUTE	VALUE
R.LVO.MORPH.001	Morphological Value	LinguisticValueType	MORPHOLOGICAL
R.LVO.SYN.001	Syntactic Value	LinguisticValueType	SYNTACTIC
R.LVO.SYN.002	Morphosyntactic	isMSValue	TRUE

RULE ID	CONCEPT	ATTRIBUTE	VALUE
	Value		
R.LVO.SYN.003	Other Syntactic Value	isMSValue	FALSE
R.LVO.SYN.004	Conjugational Value	MSValueType	CONJUGATIONAL
R.LVO.SYN.005	Declensional Value	MSValueType	DECLENSIONAL
R.LVO.SYN.006	Other Morphosyntactic Value	MSValueType	OTHER
R.LVO.SYN.007	Other Conjugational Attribute	isAgreementValue	FALSE
R.LVO.SYN.008	Agreement Value	isAgreementValue	TRUE
R.LVO.SYN.009	Other Declensional Value	isAgreementValue	FALSE
R.LVO.SYN.010	Person Value	AgreementValueType	PERSON
R.LVO.SYN.011	Politeness Value	AgreementValueType	POLITENESS
R.LVO.SYN.012	Gender Value	AgreementValueType	GENDER
R.LVO.SYN.013	Number Value	AgreementValueType	NUMBER
R.LVO.SYN.014	Aspect Value	OtherConjValType	ASPECT
R.LVO.SYN.015	Auxiliary Value	OtherConjValType	AUXILIARY
R.LVO.SYN.016	Finiteness Value	OtherConjValType	FINITENESS
R.LVO.SYN.017	Tense Value	OtherConjValType	TENSE
R.LVO.SYN.018	Mood Value	OtherConjValType	MOOD
R.LVO.SYN.019	Voice Value	OtherConjValType	VOICE
R.LVO.SYN.020	Reflexivity Value	OtherConjValType	REFLEXIVITY
R.LVO.SYN.021	Definiteness Value	OtherMSValType	DEFINITENESS
R.LVO.SYN.022	Degree Value	OtherMSValType	DEGREE
R.LVO.SYN.023	Inflection Type Value	OtherMSValType	INFLECTION
R.LVO.SYN.024	Syntactically Used As Value	OtherMSValType	USED_AS
R.LVO.SYN.025	Separability Value	OtherMSValType	SEPARABILITY
R.LVO.SYN.026	Strength Value	OtherMSValType	STRENGTH
R.LVO.SYN.027	Syntactic Dependency Value	OtherSynValType	ACTUAL_M-S_FUNCTION
R.LVO.SYN.028	Actual Morphosyntactic Function Value	OtherSynValType	PHRASE_FUNCTION
R.LVO.SYN.029	Phrase Function Value	OtherSynValType	SYNTACTIC_FUNCTION
R.LVO.SYN.030	Syntactic Function Value	OtherSynValType	SYNTACTIC_DEPENDENCY
R.LVO.SYN.031	Lexical Function	OtherSynValType	LEXICAL_FUNCTION

RULE ID	CONCEPT	ATTRIBUTE	VALUE
	Value		
R.LVO.SYN.032	Surface Sense Value	OtherSynValType	SURFACE_SENSE
R.LVO.SYN.033	Surface Polarity Value	OtherSynValType	DECLARATIVE
R.LVO.SYN.034	Interrogative Extension Value	OtherSynValType	INTERROGATIVE_EXTENSION
R.LVO.SYN.035	Deep Sense Value	OtherSynValType	DEEP_SENSE
R.LVO.SEM.001	Semantic Value	LinguisticValueType	SEMANTIC
R.LVO.SEM.002	State Of Affairs Value	SemanticValueType	EVENT
R.LVO.SEM.003	Quality Value	SemanticValueType	QUALITY
R.LVO.SEM.004	Propositional Component Value	SemanticValueType	PROPOSITIONAL_COMPONENT
R.LVO.SEM.005	Other Semantic Value	SemanticValueType	OTHER
R.LVO.SEM.006	Is Instanced Value	PropCompValueType	INSTANCED
R.LVO.SEM.007	Participant Type Value	PropCompValueType	PARTICIPANT
R.LVO.SEM.008	Semantic Role Value	PropCompValueType	SEMANTIC_ROLE

As for the *axioms* of the LVO, they have been included in Table 86. In this table, the corresponding axioms have been specified following a simplified notation, for the sake of space, expressiveness and clarity. All of them are expressed according to the usual formalism used in Mathematics and Logic for the representation of their formulae. All these axioms formalise the relationships between linguistic attributes and their corresponding values. That is, they constrain the application of the values in the LVO to those attributes in the LUO for which they are defined.

Table 86: The LVO axioms

AXIOM IDENTIFIER	AXIOM
A.LVO.SYN.01	$\forall y (Takes(hasAspect, y) \rightarrow Aspect_Value(y))$
A.LVO.SYN.02	$\forall y (Takes(usesAuxiliary, y) \rightarrow Auxiliary_Value(y))$
A.LVO.SYN.03	$\forall y (Takes(hasFiniteness, y) \rightarrow Finiteness_Value(y))$
A.LVO.SYN.04	$\forall y (Takes(hasTense, y) \rightarrow$

AXIOM IDENTIFIER	AXIOM
	$Tense_Value(y)$
A.LVO.SYN.05	$\forall y (Takes(hasMood, y) \rightarrow Mood_Value(y))$
A.LVO.SYN.06	$\forall y (Takes(hasVoice, y) \rightarrow Voice_Value(y))$
A.LVO.SYN.07	$\forall x (Reflexivity_Value(x) \rightarrow Boolean(x))$
A.LVO.SYN.08	$\forall y (Takes(isReflexive, y) \rightarrow Reflexivity_Value(y))$
A.LVO.SYN.09	$\forall y (Takes(hasPerson, y) \rightarrow Person_Value(y))$
A.LVO.SYN.10	$\forall y (Takes(hasPolitenessMarkedness, y) \rightarrow Politeness_Value(y))$
A.LVO.SYN.11	$\forall x,y ((Gender_Attribute(x) \wedge Gender_Value(y)) \rightarrow Takes(x, y))$
A.LVO.SYN.12	$\forall x,y ((Number_Attribute(x) \wedge Number_Value(y)) \rightarrow Takes(x, y))$
A.LVO.SYN.13	$\forall y (Takes(hasCase, y) \rightarrow Case_Value(y))$
A.LVO.SYN.14	$\forall x (Definiteness_Value(x) \rightarrow Boolean(x))$
A.LVO.SYN.15	$\forall y (Takes(isDefinite, y) \rightarrow Definiteness_Value(y))$
A.LVO.SYN.16	$\forall y (Takes(hasDegree, y) \rightarrow Degree_Value(y))$
A.LVO.SYN.17	$\forall y (Takes(hasInflectionType, y) \rightarrow Inflection_Type_Value(y))$
A.LVO.SYN.18	$\forall y (Takes(isSyntUsedAs, y) \rightarrow Syntactically_Used_As_Value(y))$
A.LVO.SYN.19	$\forall x (Separability_Value(x) \rightarrow Boolean(x))$
A.LVO.SYN.20	$\forall y (Takes(isSeparable, y) \rightarrow Separability_Value(y))$
A.LVO.SYN.21	$\forall x (Strength_Value(x) \rightarrow Boolean(x))$

AXIOM IDENTIFIER	AXIOM
A.LVO.SYN.22	$\forall y (Takes(isStrong, y) \rightarrow Strength_Value(y))$
A.LVO.SYN.23	$\forall y (Takes(isStrong, y) \rightarrow Syntactic_Dependency_Value(y))$
A.LVO.SYN.24	$\forall y (Takes(hasActualMorphosyntacticFunction, y) \rightarrow Actual_MorphoSyntactic_Function_Value(y))$
A.LVO.SYN.25	$\forall y (Takes(hasSyntacticFunction, y) \rightarrow Phrase_Function_Value(y))$
A.LVO.SYN.26	$\forall y (Takes(hasSyntacticDependency, y) \rightarrow Syntactic_Function_Value(y))$
A.LVO.SYN.27	$\forall y (Takes(hasLexicalFunction, y) \rightarrow Lexical_Function_Value(y))$
A.LVO.SYN.28	$\forall y (Takes(hasSurfaceSense, y) \rightarrow Surface_Sense_Value(y))$
A.LVO.SYN.29	$\forall y (Takes(hasSurfacePolarity, y) \rightarrow Surface_Polarity_Value(y))$
A.LVO.SYN.30	$\forall y (Takes(hasInterrogativeExtension, y) \rightarrow Interrogative_Extension_Value(y))$
A.LVO.SYN.31	$\forall y (Takes(hasDeepSense, y) \rightarrow Deep_Sense_Value(y))$
A.LVO.SEM.01	$\forall x, y ((Semantic_Unit_Attribute(x) \wedge Semantic_Unit_Value(y)) \rightarrow Takes(x, y))$
A.LVO.SEM.02	$\forall x ((Semantic_Value(x) \wedge \neg (Participant_Type_Value(x) \vee Semantic_Role_Value(x))) \rightarrow Boolean(x))$
A.LVO.SEM.03	$\forall x, y ((State_Of_Affairs_Attribute(x) \wedge State_Of_Affairs_Value(y)) \rightarrow Takes(x, y))$
A.LVO.SEM.04	$\forall x, y ((Quality_Attribute(x) \wedge Quality_Value(y)) \rightarrow Takes(x, y))$
A.LVO.SEM.05	$\forall y (Takes(isInstanced, y) \rightarrow Is_Instanced_Value(y))$

AXIOM IDENTIFIER	AXIOM
A.LVO.SEM.06	$\forall y (Takes(hasParticipantType, y) \rightarrow Participant_Type_Value(y))$
A.LVO.SEM.07	$\forall y (Takes(hasSemanticRole, y) \rightarrow Semantic_Role_Value(y))$
A.LVO.SEM.08	$\forall x, y ((Other_Semantic_Attribute(x) \wedge Other_Semantic_Value(y)) \rightarrow Takes(x, y))$

Thus far, all the concepts in the LVO have already been presented. They formalise the different types of linguistic values that the linguistic attributes contemplated within OntoTag can take. Also their associated (ontological) attributes, *ad hoc* relations, rules and axioms have been presented up to this point. To conclude the presentation of the LVO, some numeric statistics are shown in the next subsection about the components that it includes.

4.1.5.5 THE LVO STATISTICS

To conclude this section, the Linguistic Value Ontology statistics have been summarised in Table 87.

Table 87: The LVO statistics

LVO	CONCEPTS	INSTANCES	ATTRIBUTES	RELATIONS			RULES	AXIOMS	TOTAL
				TAX.	MERON.	AD HOC			
TOP-LEVEL	6	2	0	5	0	0	0	0	13
(MORPHO) SYNTACTIC LEVEL	38	153	38	34	0	0	38	31	332
SEMANTIC LEVEL	9	37	10	9	0	0	10	9	84
TOTAL	53	192	48	48	0	0	48	40	429
				48					

Hence, also the Linguistic Value Ontology (LVO) has already been described. This ontology, as commented above, contains the formalisation of the linguistic values that the linguistic attributes contemplated within OntoTag can take. The presentation of this ontology concludes the description of the different ontologies developed as the OntoTag model was devised. The overall statistics associated to OntoTag's ontologies have been summarised in Table 88.

Table 88: Overall statistics associated to OntoTag's ontologies

OntoTag's Ontologies	CONCEPTS	INSTANCES	ATTRIBUTES	RELATIONS			RULES	AXIOMS	TOTAL
				TAX.	MERON.	AD HOC			
TOP-LEVEL	72	22	59	38	14	15	0	7	227
(MORPHO) SYNTACTIC LEVEL	179	203	211	140	4	11	211	48	1007
SEMANTIC LEVEL	190	66	195	134	0	28	195	14	822
TOTAL	441	291	465	312	18	54	406	69	2056
				384					

The following sections (and chapters) explain how OntoTag's ontologies were applied in the formulation of OntoTag's abstract annotation architecture and scheme, as well as in the development of their respective implementations (*i.e.*, in OntoTagger).

4.2. ONTOTAG'S ABSTRACT ANNOTATION ARCHITECTURE

As already mentioned, the OntoTag model includes an abstract annotation architecture for the linguistic annotation of documents (mostly web pages) at different levels. This annotation architecture aims at (a) enabling the integration and interoperation of several linguistic annotation tools operating at the same or different levels of linguistic description; (b) comparing and combining their results for their improvement and interoperation. This annotation architecture is introduced and described in this section from an abstract and holistic point of view. The most specific implementation details are discussed in Chapter 5, where OntoTagger, an implementation of the model, is shown. However, before describing OntoTag's annotation architecture in detail, some preliminary remarks must be made.

First, as commented in the restrictions of the previous chapter, there are currently some linguistic tools for the morphosyntactic, the syntactic and, to some extent, also for the semantic annotation of texts and corpora. However, when OntoTag was being developed, no tool for their annotation at the discourse and the pragmatic levels could be found (or, at least, not for Spanish). Accordingly, OntoTag (and, hence, its annotation architecture) has been restricted to the morphosyntactic, the syntactic and the semantic levels of linguistic annotation. However, OntoTag has been devised to be easily extended to other levels of annotation, such as the discourse or the pragmatic ones.

Second, since OntoTag is an abstract model, there can be several different particular implementations of it. Each particular implementation of OntoTag, that is, each instance of this abstract architecture, will be referred to as an *OntoTag's architecture configuration*, or, simply, an

OntoTag's configuration. The original (and rather ideal) configuration of OntoTag's architecture is shown in Figure 9.

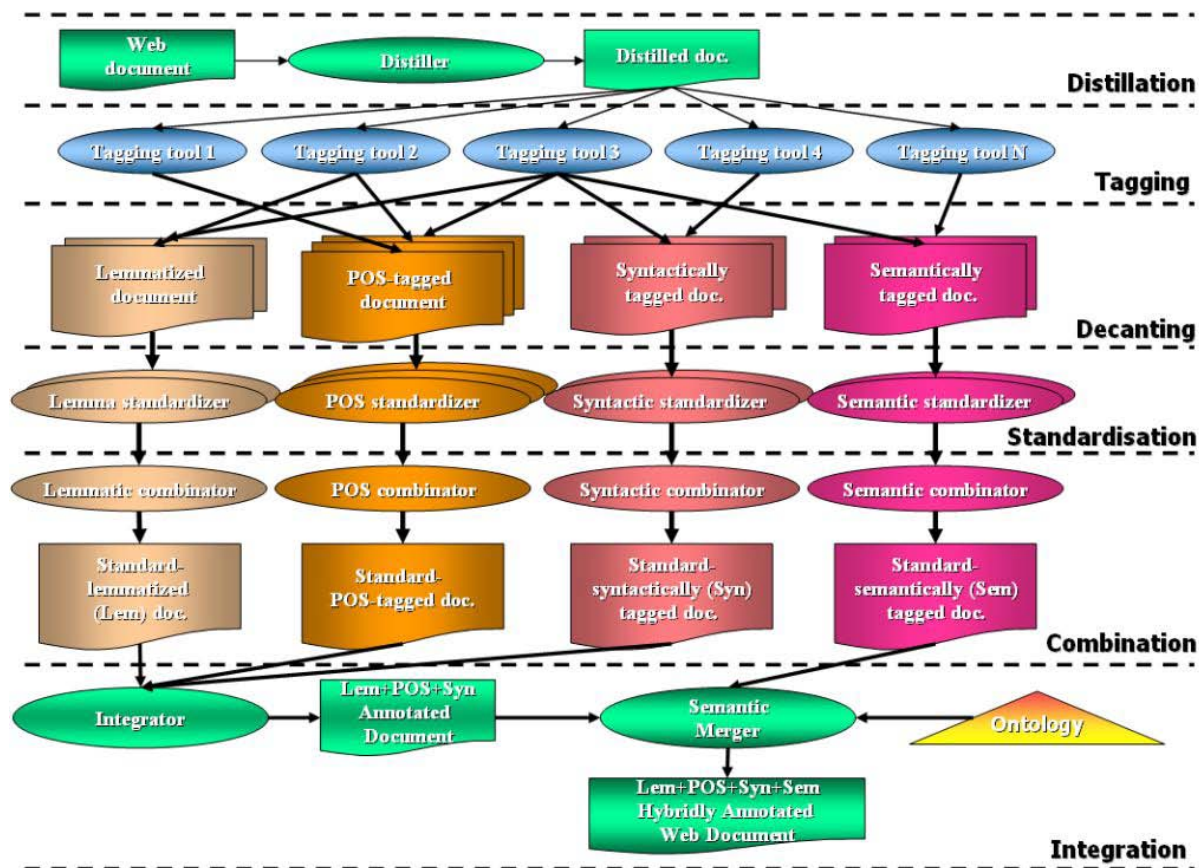


Figure 9: OntoTag's original (and ideal) architecture (configuration)

This was the first configuration proposed as an architecture for the model OntoTag. However, it was refined as OntoTagger, its first implementation, was being built. Hence, OntoTag's architecture incorporates all the lessons learned while developing OntoTagger. It evolved into a final and most general configuration, which constitutes OntoTag's (abstract) architecture for annotation. It is shown in Figure 10 (on page 185). This abstract architecture has been designed taking into account that some of its modules (whether phases, processes or tasks) might finally appear or not in each of its particular implementations. Their appearance will depend on the different tools that are to be assembled into the architecture. How it evolved during the phase of experimentation is explained all throughout the present section. In addition, in some subsections, a discussion about alternative, most frequent or more efficient configurations will also be included.

Thus, to start with, OntoTag's architecture for annotation consists of several phases of processing. Each of these phases might be subdivided into some different and rather independent processes or tasks. In other words, OntoTag's proposes to annotate each input document incrementally, as it is

processed by each of these phases, processes and/or tasks. The final aim of this architecture is to offer an automatic, standardised, high quality annotation up to the Semantic Level of linguistic description. One of the most desirable properties of this annotation is that it can be used as a sound basis for the annotation at the upper levels (the discourse-related and the pragmatic). Another most desirable property that these annotations must fulfil is that they are suitable for the Semantic Web, *i.e.*, that they are formulated by means of and/or with reference to ontological terms and using Semantic Web standard languages (such as RDF(S) and OWL).

Briefly, the five different phases of OntoTag’s architecture are (1) distillation, (2) tagging, (3) standardisation, (4) decanting, and (5) merging. Yet, this last phase is sub-divided into two intertwined sub-phases: combination, or intra-level merging, and integration, or inter-level merging¹³⁸.

Each of these phases and sub-phases of OntoTag, as well as their corresponding modules, is described next, in a dedicated subsection. Then, the resulting, most general and realistic configuration that constitutes OntoTag’s architecture is presented at the end of this section.

4.2.1. PHASE 1 – DISTILLATION

Most linguistic annotation tools do not recognise formatted (marked-up) text as input for annotation; they only accept pure, clean text to be annotated. However, the present model should be suitable for the Semantic Web, and hence, it is assumed that its input are (Semantic) Web pages, or, in general, web documents. Unfortunately, web documents rarely consist of unformatted (or non-marked-up) text. Hence, the textual information conveyed by these web documents will often have to be distilled (extracted) before using it as input for an already existing linguistic annotation tool. Accordingly, in the phase of distillation their mark-up and graphical information is removed and stored away for (1) the final re-construction of the document, or (2) the use of this meta-textual information in the following steps of the annotation process.

Thus, summing up, the input of this phase is the web document to be annotated, and its output is an unformatted document, consisting of only the textual information (the distilled, plain or clean text) of the web document which needs to be annotated.

¹³⁸ Most of the terms applied to the different annotation phases of OntoTag’s architecture have been coined by means of metaphors relating the process of language analysis and annotation to the process of chemical analysis.

4.2.2. PHASE 2 – TAGGING

As commented in the previous chapter, it is assumed that all the tools integrated in the architecture accept (at least) a plain text document as input. Accordingly, in the phase of tagging, the plain text document produced in the distillation phase is inputted to the different annotation tools assembled into the architecture. It does not matter at this point the level(s) or the format of the output annotations – it is left to the remaining phases to cope with these issues.

Thus, briefly, the input of this phase is the unformatted document, consisting of only the textual information (plain text) generated in the phase of distillation, and its output is a set of documents, one for each tool assembled into the architecture. After this phase, these documents will be tagged or annotated (i) at a certain (set of) level(s); and (2) according to a tool-dependent annotation scheme and tagset.

4.2.3. PHASE 3 – STANDARDISATION

In order for the annotations coming from the different linguistic annotation tools to be conveniently compared and combined, they must be first mapped onto a standard or guideline-compliant –that is, standardised– type of annotation, such as EAGLES (1996a; 1996b), so that:

- The annotations pertaining to the same tool but to different levels of description are clearly structured and differentiated (or decanted, in OntoTag’s terminology).
- All the annotations pertaining to the same level of description but to different tools use a common vocabulary to refer to each particular phenomenon described by that level.
- The annotations pertaining to different tools and different levels of description can be easily merged later on in a one and unique overall standardised annotation for the document being processed.

It is at this point where OntoTag’s ontologies play a crucial role. They have been developed following the existing standards, guidelines and recommendations for annotation (see Section 4.1 for a detailed description). Accordingly, annotating with reference to OntoTag’s ontologies produces a result that uses a standardised type of tagset. Thus, the tagsets and the annotations from each and every tool are mapped onto the terms of OntoTag’s ontologies. Hence, after this phase has been applied, all the phenomena being tagged share a common and standardised vocabulary for their description. In addition, this vocabulary can also be considered formal and fully semantic, since it is referred to ontologies. The level-driven, taxonomical and relational structure of OntoTag’s ontologies is also right and proper for (1) structuring and distinguishing the information into different levels; and (2) summing

up and interconnecting all of them later on again, by means of the relations already described in the ontologies themselves.

However, as commented above, the main contribution of this phase to the whole architecture is that it enables the model to handle the annotations from any tool, irrespective of the level(s) to which they pertain and the schemes (or the tagsets) employed for their generation. After the document being annotated is processed in this phase, the annotations for the same phenomenon coming from all the tools will follow the same scheme and will be, thus, comparable. A major drawback of including this phase, though, is that it requires a prior study of the output scheme and the tagsets of each of the tools assembled into the architecture. Indeed, their interpretation and mapping onto the standardised tagset obtained from OntoTag's ontologies could not be automatically determined *a priori*. Consequently, an *ad-hoc*, tool dependent standardising wrapper must be implemented for each linguistic annotation tool assembled into the architecture.

In brief, the input for this phase is the set of documents previously tagged, in the phase of tagging, according to a tool-dependent tagset and scheme (one for each tool assembled into the architecture). The output of this phase is another set of documents, differing from the input ones in that they are tagged according to a standardised, tool-independent tagset and scheme (still, one document for each tool assembled into the architecture).

4.2.4. PHASE 4 – DECANTING

A number of the linguistic annotation tools assembled into the architecture might tag at more than just one level of linguistic description. The annotations pertaining to the same tool but to a different level have to be decanted (that is, separated according to their levels and layers or types) in a way that

1. the process of the remaining phases is not complicated, but rather
2. the comparison, evaluation and mutual complement of the results offered at the same level by different tools is simplified, and
3. the different decanted results can be easily re-combined, after they have been subsequently processed.

Criterion 1 appeared as the following phase (the merging phase) and, more concretely, its combination sub-phase, was being implemented. At first glance, it would seem that fulfilling Criterion 2 would be attained by means of a straightforward decanting, consisting of the simple separation of the annotations according to their level. This means that, for each tool, one document for the lemmas, one for the rest of the morphosyntactic tags, one more for the syntactic annotations, and another one for the semantic annotations would be generated. However, it was shown empirically that a more rational

separation of levels is required, so as to simplify not only the re-combination of all the annotations, as postulated by Criterion 3, but also the process of the remaining phases, as indicated in Criterion 1. The underlying problems that call for a more rational partition of levels are explained in detail in the combination sub-phase of the merging phase (see Section 4.2.5.1). The solution to these problems was determined empirically as well, and affected the present level (decanting) with regard to how the annotations have to be partitioned and separated. Accordingly, for each annotated document coming from the tagging phase (that is, for each of the outputs of the linguistic annotation tools assembled into the architecture), a set of up to four documents has to be generated¹³⁹:

- one document containing both the lemmas and the grammatical category tags (L+POS),
- one consisting of the grammatical category tags and the morphological annotations (POS+M),
- one more for the syntactic counterpart of the annotations (Syn), and
- another one for the semantic annotations (Sem).

There remains, though, an issue that needs to be discussed: why the decanting phase is placed after standardisation? Wouldn't it be more modular and easier (at least, from a conceptual point of view) to decant the different levels and, then, standardise all of them separately? An accurate answer to these questions was determined also empirically, after some experiments were done in order to establish the best ordering for these two phases. In fact, at the beginning, these two phases had the opposite ordering (first, decanting and, then, standardisation). The problem found was that some tools might compress somehow their annotations, tagging more than one `Token` or `Morphosyntactic Unit` as a single one, for example, whilst some others might not. The solution to this problem involved standardising (at least) the morphosyntactic and the syntactic annotations before decanting. Otherwise, multiple re-numberings would have had to be done afterwards, slowing down the process of annotation unnecessarily. That is why the decanting phase came to be placed after standardisation.

Hence, in short, the input for this phase is the set of tagged and standardised documents coming from the phase of standardisation (one for each tool assembled into the architecture), and its output is a set of sets of documents (one set of documents per tool). The set for each tool contains as many documents as levels annotated by the tool (up to four documents, as a maximum, following the partition strategy mentioned above).

¹³⁹ The final number depends on the level(s) of the annotations performed by each tool. The more levels that are annotated the more documents that have to be generated.

4.2.5. PHASE 5 – MERGING

At this point, all the standardised and decanted annotations have to be merged in order to yield a unique, combined and multi-level (or multi-layered) annotation for the original input document. This last phase of annotation was intensively evaluated, revised and modified as OntoTagger was being implemented. In effect, it is the most complex part of the model, since it is responsible for two different tasks:

1. uniting (combining) all the annotations that belong to the same level, but come from different tools,
2. summing up and interconnecting the annotations that belong to different levels so as to bear a combined, integrated and unique set of annotations for the original input document.

As commented above, these two tasks are conceptually different and, thus, are considered two different (but intertwined) sub-phases in the architecture. These two sub-phases, namely combination and integration, are described in the following two subsections. The sequence of application of each sub-phase is not a straightforward one and requires a detailed explanation, which is given in Subsection 4.2.5.3.

4.2.5.1 SUB-PHASE 5.1 – COMBINATION (INTRA-LEVEL MERGING)

In this sub-phase, as indicated above, all the annotations that come from the different tools assembled into the architecture and belong to the same level are united. Therefore, the result of this sub-phase is a unique, combined and (possibly) improved annotation per level for the document being annotated. In this description of its result,

- *Unique* does not imply that the information about the sources of a particular tag be simply removed. Far from this, what is proposed here is that traces of all annotations from the different tools be also kept and attached (conveniently labelled) to the combined annotation produced by the whole architecture.
- *Improved* stands for the fact that it is expected to correct (part of) the errors introduced by one tool assembled into the architecture when tagging the document, by contrasting and complementing its results with the ones of other tool(s) also assembled into the architecture.

We postulate here a straightforward method for successfully combining each and every one of the different levels described in the previous phase (decanting). It was followed and conveniently tested

and evaluated in OntoTagger, the first implementation of OntoTag (see Chapter 5). It consists of five steps:

- Step 1 : **Tool weakness and strength analysis:** It must be studied in detail the behaviour of the different tools being assembled over a corpus sample, in order to determine their weaknesses and strengths when tagging. Statistics must be calculated about (1) the type and number of tagging failures (errors, gaps¹⁴⁰ and underspecifications¹⁴¹) of each tool; and (2) the cases in which one tool outperforms somehow the others, in order to shed some light on the upcoming steps of processing.
- Step 2 : **Error context analysis:** Much of the errors, gaps and underspecifications introduced by a tool when tagging can be corrected by simply having a look at their linguistic contexts. For instance, in Spanish, simple agreement rules for deducing the gender of a *Morphosyntactic Unit* from the gender of the preceding (and/or the following) one, can improve significantly the accuracy of the combined morphological annotation. Therefore, supplementary statistics about the failures found must be calculated, taking into account this sort of contextual information.
- Step 3 : **Modelling:** From the results obtained in the previous steps, it must be designed an abstract model that represents accurately the behaviour of the tools at the levels analysed. This model has to include heuristics for the correction of the (types of) failures found or, at least, for the most common ones. It does not need to be a very formal one, but it has to be detailed enough to allow for a suitable and precise implementation in the following steps. In effect, it is to serve as the specification for the combination module at the level being considered.
- Step 4 : **Combination method election:** The best method must be chosen for the combination of the annotations at the level under study (production rules –with or without probabilistic triggering factors–, Bayesian networks, etc.). This choice will be made according to the model generated in the previous step (in particular, according to the proposals for solving the failures detected).
- Step 5 : **Incremental tuning:** A prototype of the combination module has to be developed, including the implementation of a gradually growing set of failure-correcting heuristics already modelled in Step 3. This prototype has to be double-validated after the inclusion of a new set of heuristics. This double validation shall be made against the corpus sample before reaffirming the inclusion of the set of heuristics implemented into the prototype:

¹⁴⁰ Phenomena and or elements not annotated, even though they should.

¹⁴¹ Elements whose annotation can be more precise, further subspecified or characterised.

- a. It must be checked whether the new set of heuristics introduces new (types of) failures or not. Should it introduce new failures, then it will be removed or inhibited, and a revision and adjustment (tuning) of the model will be required (Step 3).
- b. If no new failures appear, it is also necessary to check whether the new set of heuristics does in fact eliminate the failure it is supposed to correct. If the failure is eliminated, then the new release of the prototype, including this last set of heuristics, is reaffirmed, and a new iteration of Step 5 is executed until the last failure-correcting heuristic modelled has been implemented. Or else, if the new set of heuristics does not correct the failure, then a revision and adjustment (tuning) of the model is required (Step 3), so as to determine whether (i) a new heuristic must be added to completely eliminate the failure, if the last one worked only for certain cases, and needs complementing; or (ii) it simply must be substituted by a more accurate one.

According to the decomposition of the tagged documents made in the decanting phase, the combination sub-phase consists (at most) of four different combination processes, one for each type of document that is generated in the previous phase: L+POS combination, POS+M combination, syntactic combination and semantic combination. Each of them is described below from an abstract and general point of view in a dedicated subsection. A detailed description of the most common errors, gaps and underspecifications at each level, as well as their correction proposals, can be found in the chapter dedicated to the implementation of *OntoTagger* (Chapter 5).

4.2.5.1.1 L+POS Combination

In this process, all the *Lemma* and morphosyntactic annotations (L+POS annotations) coming from the different tools assembled into the architecture are combined together. The aim of this process is, therefore, to produce a unique and improved L+POS annotation for the document being annotated, following the general combination method aforesaid. Briefly, the main goal of L+POS combination is threefold:

1. Tagging correctly as many morphosyntactic units as possible with a part-of-speech (POS) tag.
2. Mapping these POS tags onto their corresponding concepts within *OntoTag*'s Linguistic Unit Ontology (LUO).
3. Tagging correctly as many morphosyntactic units as possible with a *Lemma* tag.

The components (1) and (3) of this threefold goal are closely interrelated: the *Lemma* of a *Morphosyntactic Unit* can be determined only after its part-of-speech has been established. For example, the Spanish word 'compras' (\approx buyings / (you) buy) can be lemmatised as 'compra' (buying) or 'comprar' (\approx buy), depending on the part-of-speech that this *Morphosyntactic Unit* has in its

context: respectively, Verb (V) or Noun (N). Whenever a wrong POS tag is assigned to a Morphosyntactic Unit, a wrong Lemma tag is assigned to the Morphosyntactic Unit in question as well. Here are two different examples of use of ‘compras’, extracted from the web:

- a) *En los tres primeros meses del año, las **compras** subieron un 28%.¹⁴²*
- b) *Si **compras** en Internet, no olvides leer la letra pequeña.¹⁴³*

If ‘compras’ were assigned an incorrect POS tag, that is, a Verb (V) POS tag in Example (a), or a Noun (N) POS tag in Example (b), it would be assigned also a wrong Lemma tag (‘comprar’ in Example (a) and ‘compra’ in Example (b)).

Since failures at the lower levels are propagated and magnified at the upper levels, it is important to reduce the amount of tagging failures at the lower levels and, more specifically, the morphosyntactic tagging failures. A detailed description of the most common morphosyntactic tagging failures, together with their correction proposals, can be found in Subsection 5.3.2.2 (implementation of the L+POS combination sub-phase in OntoTagger).

4.2.5.1.2 POS+M Combination

The POS+M combination process is in charge of uniting all the morphological annotations coming from the different tools interoperating in the architecture. Therefore, the aim of this process is to produce a unique and improved morphological annotation for the document being annotated, following the general combination method aforesaid. For this purpose, the improved POS tagging output of the L+POS combination process is used as an additional input to this process, which can be subdivided into the following sub-processes:

1. To begin with, all corrections performed at the morphosyntactic (POS) Level by the L+POS combination process must be reaffirmed and propagated to the rest of levels in this phase.
2. Next, on the basis of the POS tags assigned to the morphosyntactic units in the document, the morphological attributes included in EAGLES (1996a) for each (type of) morphosyntactic unit(s) must be extracted from OntoTag’s Linguistic Attribute Ontology (LAO). These standardised morphological attributes must be compared to the ones actually included in the morphological annotations of each tool. On the hand, some attributes might be spurious, if an erroneous POS was assigned previously. In this case, they need to be substituted by the right ones after its correction within the L+POS combination sub-phase. On the other hand, they

¹⁴² During the first three months of the year, the **buy** numbers increased a 28 percent.

¹⁴³ If you **buy** on Internet, don’t forget to read the small print.

might not appear in these annotations due to tagset gaps or underspecifications, and require being introduced at this stage.

3. After that, the morphological values associated to each of these morphological attributes in EAGLES (1996a) must be extracted from OntoTag's Linguistic Value Ontology (LVO). In effect, it is necessary to (i) check whether the values included in the annotations of each tool agree with the standardised values of the LVO, and (ii) map the former onto the latter.
4. Then, a combined morphological annotation must be generated. It will be obtained by means of the morphological combination module of a prototype that implements the correction model designed for this sub-phase. Hence, in this combined morphological annotation, as many morphological annotation failures (errors, gaps and underspecifications) as possible will have been removed.

As in the case of lemmatisation, it must be remarked that the attributes for a *Morphosyntactic Unit* can be determined only after its part-of-speech has been established. As shown above, the *Morphosyntactic Unit* 'compras', can be assigned either a Noun (N) or a Verb (V) POS tag. Accordingly, it could be tagged with the following morphological attributes: (a) Gender, Number and Case (for nouns); and (b) Person, Gender, Number, Finiteness, Mood, Tense, and Voice (for verbs). A detailed description of the most common annotation failures at this level and their correction proposals can be found in the implementation of the POS+M combination sub-phase in OntoTagger (Subsection 5.3.3).

4.2.5.1.3 Syntactic Combination

The syntactic combination process is in charge of uniting all the syntactic annotations coming from the different tools assembled into the architecture. In other words, the aim of the syntactic combination process is to produce a unique and improved syntactic annotation for the document being annotated, following the general combination method aforesaid. This process can be subdivided into the following four sub-processes:

1. First of all, all the different decanted documents being combined at the *Syntactic Level* have to be scanned in parallel for morphosyntactic gaps. Clearly, it is easier to detect this type of failures at this level, by comparing the tokens (morphosyntactic units) contained in each of these documents. All the information about POS tagging is also an input to this process, since POS tags constitute the interface between the morphological and the syntactic levels. In the Phrase-Structure Grammar theory, for example, the morphosyntactic information is required to build the syntactic tree that represents the syntactic structure of an utterance. The leaves of such a syntactic tree are made out of morphosyntactic information. At the same time, the points

at which each tool introduced an end of sentence or paragraph tag might differ and, thus, they must be compared and unified as well. Thus, at this stage, any discontinuity or incoherence in the numbering of the tokens across the different documents is found and solved, by applying the syntactic heuristics included in the correction model for this process (see Section 5.3.1).

2. Next, annotations pertaining to the same syntactic phenomena but coming from different tools are conveniently combined together. Special attention must be paid in this sub-phase to finding, filling and labelling, in the predicates being annotated, both (a) the different syntactic slots (parameters) not filled yet; and (b) the dependencies still not tagged. This must be done in order to complete the syntactic structure of these predicates.
3. Then, all the different layers of syntactic annotation must be united, combined and interrelated in a one and only document with the overall conjoined syntactic annotation for the web document being annotated. All the syntactic units in this conjoined syntactic annotation must be mapped onto their corresponding concepts within the LUO. In addition, all the syntactic dependencies and relationships that might hold between these syntactic units must be mapped onto their corresponding terms in OntoTag's ontologies. All of their associated attributes must be mapped onto their corresponding terms in the LAO afterwards. And finally, all their values in the annotated documents must be mapped onto their corresponding terms in the LVO.
4. Finally, all the corrections performed at the (Morpho)Syntactic Level by the syntactic combination process must be propagated to the rest of decanted documents (levels), so that their corresponding units can be re-numbered and re-adjusted consistently.

These four different sub-processes of syntactic combination might be split for the sake of efficiency if necessary. They can also be interleaved with or put ahead of other (apparently) preceding combination tasks when required. For example, searching for the morphosyntactic clashes between annotations coming from different tools from a syntactic point of view helps solving them by the addition of some contextual (syntactic) information, available at this level.

Another interesting consideration is that the unification of sentence and paragraph endings (Sub-phase 4.2.5.1.3) should be done as soon as possible, since it reduces the amount of token re-numberings required to have the document annotated. Consequently, it also reduces significantly the amount of time required to annotate a document. This was the case in the implementation of OntoTagger. Hence, a detailed description of this phenomenon and of the most common annotation failures at the Syntactic Level and their correcting heuristics can be found in Section 5.3 and in Subsection 5.3.1 (implementation of the syntactic combination sub-phase in OntoTagger).

4.2.5.1.4 Semantic Combination

The semantic combination process is in charge of uniting all the semantic annotations coming from the different tools assembled into the architecture. Thus, it aims at producing a unique and improved semantic annotation for the document being annotated, following the general combination method aforesaid. This process can be subdivided into the following sub-processes:

1. The annotations coming from different tools but pertaining to the same layer of the `Semantic Level` (see Subsection 3.2.1.3.1 for details about the different layers of semantic annotation), must be conveniently combined together.
 - a. Semantic combination, as it happens with the rest of levels of combination, and particularly in its layers of named entity, sense and semantic field tagging (or annotation), is simplified by the standardisation phase. In fact, all the named entities, senses and semantic fields appearing as tags will refer to the same (`OntoTag`) ontology (or ontologies). In spite of this, the combination in these layers might not be straightforward, especially due to word sense ambiguities, and might require using both WSD (Word Sense Disambiguation) and Ontology Mapping techniques.
 - b. Special attention must be paid also in this sub-phase to finding, filling and labelling the different semantic slots (arguments, participants and/or actants) not filled yet in the predicates being annotated, in order to complete the semantic structure of these predicates.
2. All the different layers of semantic annotation must be united, combined and interrelated in a one and only document. This document will contain the overall conjoined semantic annotation for the (web) document being annotated.

To conclude, a detailed description of the most common annotation failures at the `Semantic Level` and their correcting heuristics can be found in Section 5.4 (implementation of the semantic combination sub-phase in `OntoTagger`).

4.2.5.2 SUB-PHASE 5.2 – INTEGRATION (INTER-LEVEL MERGING)

In the sub-phase of integration, as mentioned before, the annotations at all levels are summed up and interconnected again, after being (1) standardised and (2) combined, if more than one linguistic tool assembled into the architecture annotate the input document at the same level. All of these annotations, belonging to different levels, are united to bear a standardised, integrated, unique, and

(possibly) improved overall annotation for the input document¹⁴⁴. This overall annotation will observe and comply with the restrictions included in OntoTag's annotation scheme, explained in Section 4.3.

This seems a rather straightforward task. However, the sequencing and interleaving of both combination and integration requires a bit of operational research in order to save time and avoid re-executing tasks unnecessarily when merging the different annotations. These rather thorny issues are discussed in the next subsection.

4.2.5.3 PUTTING COMBINATION AND INTEGRATION TOGETHER

At the beginning, a rather naïve approach for merging the annotations at all levels was proposed: in the sub-phase of combination, it was assumed that each `Linguistic Level` could be handled separately. This led to a paralleled architecture for this sub-phase, where each level was combined independently and at the same time as the others. The experimentation showed that this could not be carried out in that manner, since

1. The POS tag attached by a certain tool to a certain `Morphosyntactic Unit` might differ from the one(s) attached to it by other tool(s). Due to the fact that the `Lemma` and the morphological attributes of each `Morphosyntactic Unit` are related to its grammatical category (*i.e.*, its part-of-speech), its real POS tag has to be established before combining the `Lemma` or the morphological information of that unit.
2. Furthermore, in some cases, there is some structural disagreement in the manner morphosyntactic units were tagged by each tool for a given document. For example, some tools (not each and every one of them) would group several morphosyntactic units and tag them as though only one existed. More concretely, some of them would tag three consecutive periods ('...') as such, whilst others would consider them altogether a unique `Morphosyntactic Unit` and assign it a `Suspension Points` tag. Some others (again, not each and every one of them) would tear an orthographic unit apart into its morphosyntactic constituents and tag them separately. For instance, the Spanish token 'Díselo' would be divided into the three morphosyntactic units 'Di' + 'se' + 'lo', which would be tagged separately. These phenomena generated some numbering discontinuities and incoherencies between the morphosyntactic annotations coming from different tools (when compared)¹⁴⁵. These numbering discontinuities and incoherencies required being explained by means of a computational model that, after being implemented, could help solve them as well. But when solved, they motivated a kind of

¹⁴⁴ The remarks about unique and improved stated in Subsection 4.2.5.1 are also of application here.

¹⁴⁵ As experimentation showed, the following information was needed to identify every morphosyntactic unit: (i) its order of appearance in a sentence; (ii) the order of appearance of that sentence in its paragraph; and (iii) the order of appearance of that paragraph within the document being annotated.

re-numbering of the units already tagged. This re-numbering had to be propagated to the annotations of both the syntactic and the semantic levels after their decanting and before the combination sub-phase at these levels was run.

3. As mentioned above, the ends of the sentences and paragraphs generated also several conflicts at the `Syntactic Level`. They were identified and tagged differently by the linguistic tools assembled into the architecture. Hence, when contrasting and combining the syntactic annotations from these tools, their syntactic units had to be re-numbered at this level as well. This, in turn, motivated a re-numbering of the units in the other levels, also after decanting and before the combination sub-phase at these levels was run.

As an immediate consequence, it seemed clear that:

1. Morphosyntactic combination should be accomplished before tackling Lemma combination.
2. Morphosyntactic combination should be done before dealing with morphological combination.
3. Syntactic segmentation into (paragraphs and) sentences should be done before any other combination process was run, in order to achieve an accurate and consistent numbering of tokens as soon as possible or, at least, before proceeding with the merging phase.

Thus, the following ordering for the combination and integration of the different annotations was empirically determined:

1. Syntactic Combination (A): The process of syntactic combination must be split and, first of all, any discontinuity or incoherence in the numbering of the tokens across the different documents has to be found and solved (see the details in Subsection 5.3.1):
 - a. All the decanted documents being combined at the `Syntactic Level` must be scanned in parallel for numbering discontinuities and incoherencies at the `MorphoSyntactic Level`. These numbering discontinuities and incoherencies must be conveniently solved at this point.
 - b. At the same time, the locations of the ends of sentence or paragraph identified by the different tools have to be compared and unified.
 - c. This sub-process ends when (i) a final and unified numbering of the morphosyntactic units has been achieved; and (ii) all changes have been propagated to the rest of the decanted documents (at every level).
2. L+POS Combination: Next, morphosyntactic categories (POS tags) must be combined; also lemmas must be combined subsequently. Then, the results of this combination sub-phase have to be propagated to the rest of the levels and layers (morphological, syntactic and semantic).
3. POS+M Combination: At this stage, morphological attributes and values can be combined too. In some cases, this process can be executed in parallel with the following one.

4. Syntactic Combination (B): After the POS combination, the remaining tasks of the syntactic combination process can be executed as well:
 - a. Annotations pertaining to the same `Syntactic Layer` but coming from different tools are conveniently combined together.
 - b. Then, all the different syntactic layers of annotation must be united, combined and interrelated in a one and only document with the overall conjoined syntactic annotation for the (web) document being annotated.
5. Semantic Combination: Finally, the semantic annotations for the input document coming from different tools have to be combined layer by layer (possibly in parallel). This might require obtaining all the lower level annotations before executing the process of semantic combination. Thus, it is placed in the last position of integration. Actually, in most cases, semantic annotation requires, at least, a POS tagged input on top of which the semantic one is added. In addition, a certain degree of parsing (*i.e.*, syntactic annotation) seems indispensable for deeper and broader semantic analyses and annotations. As a consequence, there is little chance that this process be executed in parallel with any of the previous ones.

In summary, the input for the whole phase of merging is the tagged, standardised and decanted set of sets of documents (one set of documents per tool assembled into the architecture) coming from the phase of decanting, and its output is a one and only final document, morphosyntactically and semantically annotated. This multi-level annotation (1) incorporates the standardisation, combination and merging of the annotations coming from all of the linguistic tools integrated into the configuration; (2) is expressed by means of a Semantic Web standard languages and with reference to a set of linguistic ontologies¹⁴⁶; and, therefore, (3) is a hybrid annotation, *i.e.* linguistic and ontological, since it consists of linguistic information encoded by means of ontological terms.

To conclude, a general configuration of this block (but more realistic than the one presented in Figure 9) is shown in Figure 10 (see page 185).

After introducing OntoTag's (abstract) architecture for annotation, it is the time to present its (abstract) annotation scheme. This will be done in the following subsection.

¹⁴⁶ Thus, it is suitable for the Semantic Web and its different purposes, since it incorporates semantic information about the content of the input documents.

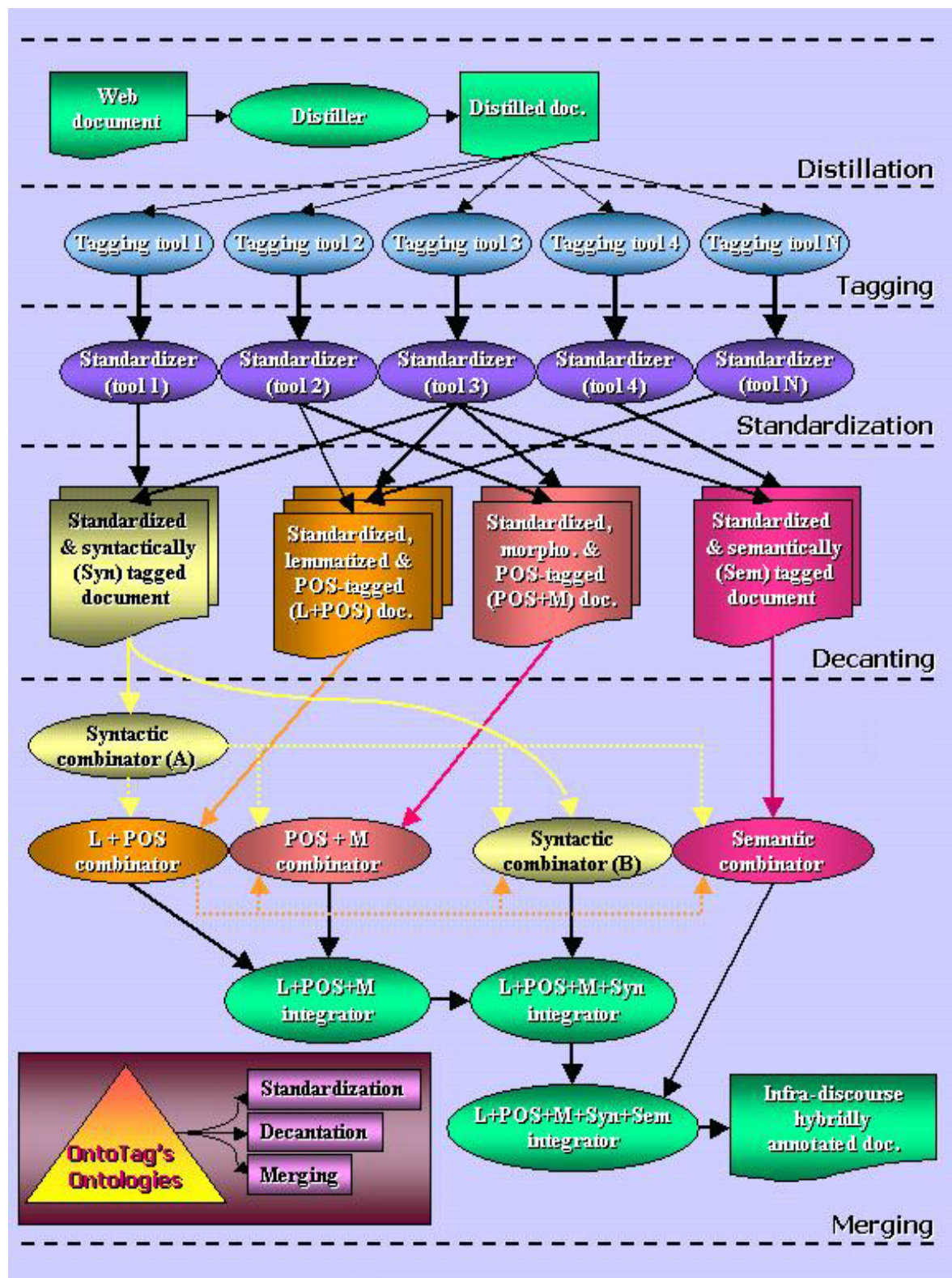


Figure 10: OntoTag's (final) architecture

4.3. ONTOTAG'S ABSTRACT ANNOTATION SCHEME

The goal of the abstract annotation scheme of OntoTag is to show (i) how the annotations of linguistic tools operating at different linguistic levels and/or layers can be summed up and interconnected; and (ii) how tool-dependent linguistic annotations can be mapped onto a sort of tool-independent annotations and, in this way, be standardised.

Therefore, the abstract annotation scheme of OntoTag tries to cover, integrate and interconnect a number of the different levels and layers of linguistic description and/or annotation developed hitherto (see Section 3.2.1, The Linguistic Approach to Annotation). Special attention has been paid to those annotations which have already been object of standardisation, or for which any kind of recommendations or guidelines have already been determined. In particular, the annotations contemplated in OntoTag are the ones mentioned in EAGLES (1996a) and shown in Figure 11: morphosyntactic annotations, lemma annotations, syntactic annotations, semantic annotations and, to a much lesser extent, also discourse annotations¹⁴⁷.

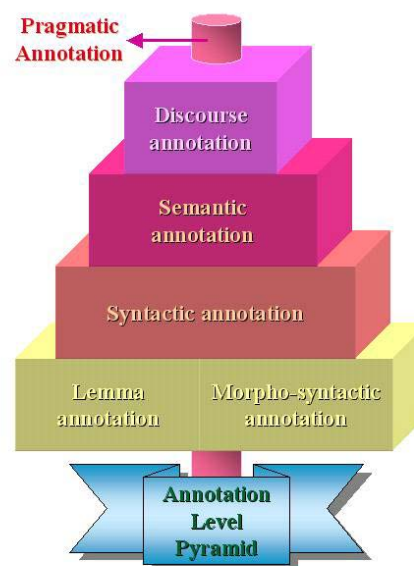


Figure 11: Main linguistic levels of annotation

With respect to oral corpora and the prosodic or the phonologic levels, they are of no real interest in the context of the web documents under consideration. Thus, these aspects have been neglected somehow, so as to focus on the other ones. So, briefly, this annotation scheme aims at the generation of a unique output document for each input document that integrates and interconnects the different types of annotations contemplated in the model (commented above).

In addition, according to the goals of the present dissertation, on the one hand, this scheme had to be both linguistically and ontologically based. In other words, the resulting annotation scheme had to (i) account for the whole set of phenomena being annotated in a systematic and unambiguous way; and (ii) use ontology terms to annotate the contents of the input document. As already explained, in order to fulfil this twofold requirement, the linguistic knowledge associated to the annotation levels

¹⁴⁷ Named entities, basically – see Section 5.4.2. As for pragmatic annotation, even though it is not mentioned in EAGLES (1996a), is also part of the linguistic knowledge that will be captured by OntoTag's ontologies. However, it has not been implemented anyhow in OntoTag's annotation scheme and architecture yet, since further, complex and long-lasting research is expected to be done before any implementation or recommendation is available for it.

contemplated within OntoTag was formalised into a set (or a network) of ontologies (presented in Section 4.1).

On the other hand, this annotation scheme¹⁴⁸ had to be suitable for the Semantic Web, that is, the resulting annotations had to be machine-readable. This entailed that they had to be represented in a formal, standardised and easy-to-validate way. This, in turn, entailed a language-oriented approach, that is, sticking to the formal language(s) that would be used for its implementation. As the implementation of OntoTagger's annotation schemas showed (see Section 5.5), this second requirement can be fulfilled if (i) OntoTag's annotation scheme is implemented by means of the standard languages usually used for semantic annotation in the Semantic Web (*e.g.*, XML, RDF(S) and/or OWL)¹⁴⁹; and (ii) the terms of the OntoTag's ontologies are included as the labels (or, equivalently, as the elements) of these annotations, as with other Semantic Web annotations. This is precisely what is proposed in the present model's annotation scheme, which is described in this section.

As OntoTag's abstract annotation architecture, OntoTag's annotation scheme evolved during and after the phase of experimentation of the present dissertation (*i.e.*, the implementation of OntoTagger). Hence, it incorporates all the lessons learned when developing OntoTagger's annotation schemas. This generated some inconsistencies between OntoTagger's annotation schemas and OntoTag's annotation scheme, which will be explained in due time.

Thus, to begin with, it was clear from the very beginning that OntoTag's annotation scheme had to be a type of **multi-dimensional markup**¹⁵⁰, combining typographic, linguistic and ontological annotations. Indeed, it had to integrate (i) the HTML typographic information of the input WWW documents; (ii) the linguistic annotations coming from the linguistic tools; and (iii) the ontological annotations generated within OntoTag. This requirement was fulfilled by (a) developing OntoTag's ontologies, which conceptualise the knowledge associated to both typographic and linguistic annotations, and (b) building the annotation scheme of OntoTag in terms of the ontologies obtained.

It was also clear that OntoTag's annotation scheme had to be a **multileveled annotation scheme** as well, since it had to cover several levels of annotation. In effect, it had to integrate at least those levels included in the annotations of the already existing linguistic annotation tools, namely the

¹⁴⁸ After an analysis of the literature available, it was observed that the term 'annotation scheme' was more frequently used within the linguistic community and that the term 'annotation schema' was more frequently used within the computational community. Henceforth, they will be used unambiguously, using the term **scheme** to refer to the (abstract) design aspects of annotations (which happen to be more linguistic) and the term **schema** to refer to the implementation details of these annotations (which happen to be more computer-related in the context of this Ph.D. dissertation).

¹⁴⁹ See Subsection 5.5, OntoTagger's Annotation Schemas.

¹⁵⁰ **Multi-dimensional markup** refers to any combination of document annotations. For example, a document may contain specifications of typographic entities (headings, paragraphs and lines) as well as annotations of linguistic entities (sentences, phrases and words). The combination of two or more of such annotation sets is called multi-dimensional markup (MDM FAQ, 2006).

morphosyntactic, the syntactic and the semantic levels. This was a salient aspect, which was immediately incorporated into the specification of the annotation scheme.

To be a multileveled annotation scheme, OntoTag's annotation scheme had to fulfil also the following requirements:

1. It had to be a sort of **standoff annotation**¹⁵¹, in order to avoid the biggest problem associated to markup combination, *i.e.*, overlap. As it is stated in the MDM FAQ (2006), there is an *overlap* in annotations when one entity continues after the end of a previously started entity. For example, the German Sentence 'Ich **ziehe** den Mantel **an** und **gehe** dich **suchen**' (≡ 'I **put** the coat **on** and I **go** and **look** for you') contains at least one annotation overlaps. In effect, there is an overlap between (1) the annotation of the conjugated form of the Verb 'anziehen' (≡ 'put on'), which is divided into 'ziehe' (≡ 'put') and 'an' (≡ 'on', in this case)¹⁵², and (2) the annotation of the Noun Phrase ('den Mantel' ≡ 'the coat'), which separates them. Such a phenomenon can easily occur if two independent annotation sets are combined, but it is not allowed in standard versions of XML. This is indeed a problem of the structure of XML/SGML, but abandoning it was not an option, since XML is the current *de facto* annotation standard. Additionally, various XML processing software systems are being developed, all of which would not be of use with OntoTag if XML were left aside. Therefore, OntoTag's annotation scheme implementations had to use a particular type of XML documents to contain the standoff annotations of the input documents.
2. In order to ease the integration of the different levels, OntoTag's annotation scheme had to follow as many international standards, guidelines and recommendations as possible. On the one hand, from a linguistic point of view, the main source of stable recommendations and guidelines came from the EAGLES (1996a; 1996b; 1996c and 1999) project¹⁵³. These guidelines were, therefore, incorporated into OntoTag's annotation scheme. On the other hand, from a computational point of view, using XML, RDF(S) and/or OWL was an obvious requirement, and this involved using its associated referencing mechanisms (Uniform Resource Identifiers –URIs– and namespaces¹⁵⁴, etc.). These languages and mechanisms, in addition, seemed most helpful at solving the problem of referencing and linking elements in a standoff

¹⁵¹ **Standoff annotation** is the most common approach to combining different text annotations (ISO/LAF, 2009). In a standoff annotation, independent annotations pointing to positions in a text file are kept aside from the data. An example of a well-known system that uses standoff markup is GATE (General Architecture for Text Engineering) (MDM FAQ, available online at <http://ilps.science.uva.nl/nlpxml2006/faq.html> [visited on 17/01/2012]).

¹⁵² And dislocated, since German grammar requires that the Verb Particle 'an' be located at the end of the Clause or the Sentence.

¹⁵³ When the model was developed, no standard for linguistic annotation had been released yet. The development of linguistic annotation standards started when OntoTag was being evaluated. Only recently have they reached an acceptable stability and degree of consensus.

¹⁵⁴ Put simply, a *namespace* is a set of names in which all names are unique. A **namespace** is a context in which a group of one or more identifiers might exist. An identifier defined in a namespace is associated with that namespace. The same identifier can be independently defined in multiple namespaces, that is, the meaning associated with an identifier defined in one namespace is independent of the same identifier declared in any other namespace. Languages that support namespaces specify the rules that determine which namespace an occurrence of an identifier (*i.e.*, not its definition) belongs to ([http://en.wikipedia.org/wiki/namespace_\(computer_science\)](http://en.wikipedia.org/wiki/namespace_(computer_science))).

annotation and, therefore, they are the languages proposed here for implementing OntoTag's annotation scheme. A clear example that uses one of these languages and implements these mechanisms is shown in Section 5.5.

Eventually, all these requirements were first met in the realisation of OntoTagger's XML Schema, (shown and described in APPENDIX A: OntoTagger's Annotation XML Schema)¹⁵⁵. OntoTagger's XML Schema describes formally the structure that OntoTag(ger)-XML annotated documents must conform to. Some graphical representations of OntoTagger's XML Schema are introduced henceforth, in due course, for the description of the design of OntoTag's annotation scheme.

The top level of OntoTagger's XML Schema (and, hence, also of OntoTag's annotation scheme) is sketched in Figure 12. This XML Schema shows the tree structure that underlies every OntoTag(ger)-annotated document, and presents the different **levels of annotation incorporated within OntoTag's annotation scheme**.

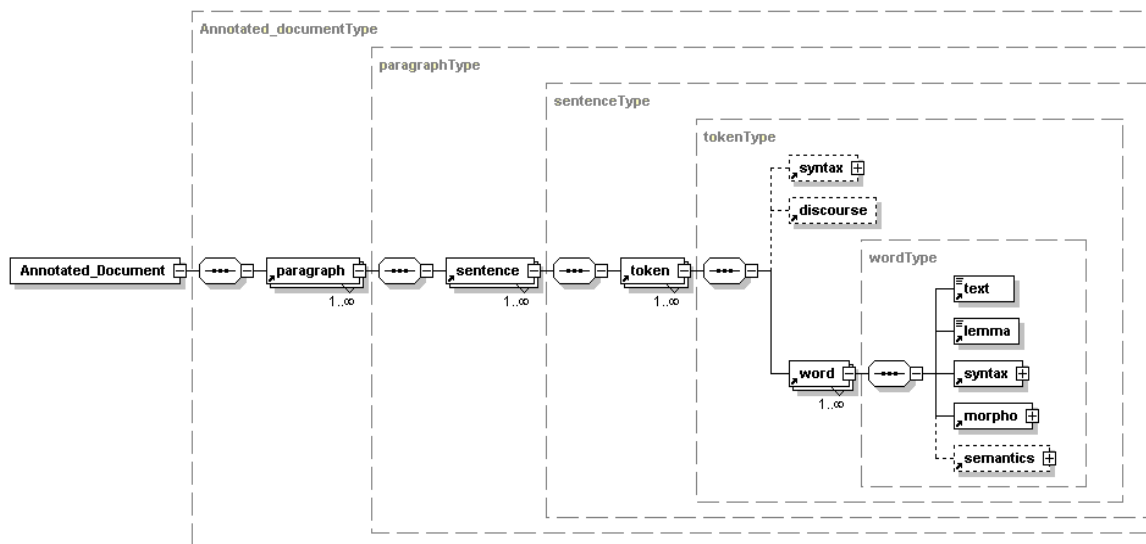


Figure 12: A graphical representation of OntoTagger's XML Schema

The main element is assigned an *Annotated_Document* label, and it might contain any number of *paragraph* elements (but at least one – see the 1..∞ indication below the *paragraph* element); a *paragraph*, in turn, can contain from one to any number of *sentence* elements (and so forth). As for the information contained in *word* elements, it includes the labels *text* and *lemma*, used to specify the word form in the input text and its lemma tagging, respectively; besides, the annotations at the syntactic and the morphosyntactic levels are nested within the *syntax* and the *morpho* elements shown

¹⁵⁵ Also an RDF Schema was developed to structure and validate the annotations obtained by means of OntoTagger. The annotations in OntoTagger-RDF annotated documents are derived from it. This RDF Schema, in turn, was used to structure the contents of the OntoTagger-OWL-annotated documents generated by the system. It has not been included here for the sake of space, but suitable examples of its application, which give an idea of its specification, are included in Subsection 5.5.2.

in the figure. The *semantic* element includes the semantic annotation of the word, but it does not always appear – this is the meaning of the dashed lines surrounding this element in Figure 12. As it can be observed in this figure as well, a *Token* can optionally contain syntactic and discourse annotations, headed by a *syntax* or a *discourse*¹⁵⁶ element, correspondingly.

As far as the **design of the tagset of OntoTag’s annotation scheme** (that is, its particular set of tags for the annotation of units, attributes and values) is concerned, the definition of a kind of EAGLES (1996a) intermediate tagset extension¹⁵⁷ was considered in the first stages of development of the scheme. However, it had to be abandoned later on, because of time restrictions. It was not a priority and, besides, it raised some problems, which required a solution not contemplated in the recommendations. The main problems found were that (i) it is hardly human-readable and (ii) in certain cases, it has not a straightforward way to be machine-read either.

In particular, in an EAGLES (1996a) intermediate tagset, those attributes whose set of values has more than ten members (*e.g.*, the attribute *Case* in Basque, which can take fifteen values – see Sagüés-Subijana (1980)) have to be represented by means of two decimal digits, while most of the remaining values need only one decimal digit for being represented. This is an issue not supported by EAGLES (1996a) recommendations. Therefore, the intermediate tagset could not be uniformly and unambiguously designed or detailed unless a complementary representation mechanism was enabled. The purpose of this representation mechanism would be to make explicit which attributes need to be coded with more than one decimal digit. Such a representation mechanism was not included in the recommendations (EAGLES, 1996a) and, hence, this solution would be language- and/or implementation-specific. Unfortunately, this is precisely what the specification of the intermediate tagset tried to solve: it is claimed to be a method of language-neutral representation. This last drawback, summed up to the rest of disadvantages of this tagset design, forced the adoption of a different tagset which is, still, EAGLES conformant¹⁵⁸.

In this final tagset, each unit, independently of its level, has its particular set of associated attributes, which take concrete values. These values, at least the morphosyntactic ones, are usually represented by means of broadly known codes or acronyms (this is also applicable to units, particularly in the case of morphosyntactic categories (EAGLES, 1996a)). Linguists are more

¹⁵⁶ However, no real discourse annotation was performed by OntoTagger. In this annotation schema, named entities, annotated within the Instance Semantic Annotation Layer of OntoTagger, were considered also units of the *Discourse Level*. This is due to the fact that they are also (co-)referential units, which are annotated in the Anaphora Resolution (or Co-Referential) Layer of the *Discourse Level*.

¹⁵⁷ This is a proposal for morphosyntactic annotation which could be extended to cover the rest of annotation levels; it recommends representing each value of an attribute with the decimal digit designating its order number within the list of values for that attribute in EAGLES (1996a) recommendations. This method for designing the tagset is claimed to be language-neutral (EAGLES, 1996a), but it seems that this claim is too strong, as it is explained above.

¹⁵⁸ EAGLES (1996a) intermediate tagset representations are not mandatory: they are only recommended. There are other ways to represent tagsets conforming to these recommendations, such as the ones eventually defined in OntoTag and implemented in OtoTagger.

accustomed to these commonly-used **acronyms**, which constitute a sort of compressed format of annotation. Hence, using them in an annotation scheme should make its annotations more readable to linguists. However, the present model aims not only at human-readability but also (and rather preferably) at machine-readability. Unfortunately, including only the acronyms in OntoTag’s annotation scheme would prevent machine-readability from being attained, provided that machine-readability is enabled mainly by the inclusion of full ontological terms in semantic annotations.

Therefore, both types of representations (i) have been enabled within OntoTag’s annotation scheme as two different non-exclusive alternatives and (ii) have also been implemented within OntoTagger’s annotation schemas. On the one hand, the commonly-used acronyms have been included as identifiers of units and values in OntoTagger’s XML Schema. On the other hand, this format of annotation was not implemented in OntoTag’s RDF- or OWL-annotated documents, where it was decided to include (only) the full terms. This difference is due to the fact that OntoTag(ger)’s XML-annotated documents are intended to be, apart from machine-readable, more human readable than OntoTag(ger)’s RDF- and OWL-annotated documents. The last two are more machine-readable-oriented.

Yet, the acronyms aforementioned are not subject to any recommendation, guideline or standard, not even at the `MorphoSyntactic` Level. Certainly, some of them can be considered a *de facto* standard (such as M for Masculine or S for Singular). However, a systematic and easy to standardise **method for choosing acronyms** (both for units and for attributes values¹⁵⁹) had to be formulated. The proposal made within OntoTag’s annotation scheme is the following¹⁶⁰:

1. Listing the different values for an attribute (*id.* the units at a given level with the same subcategorisation degree) in English, in alphabetical order;
2. Determining which of these values (*id.* units) have a recommended or standardised tag to describe them, and assigning them this particular tag (*e.g.*, Noun (N) or Noun Phrase (NP));
3. Choosing a suitable tag for the rest of values (or units) as follows:
 - a. While there remains some value (*id.* unit) that has not been assigned an acronym:
 - i. The first value (*id.* unit) in the list to which no acronym has been assigned yet is taken.
 - ii. The first letter of the English term for this value (*id.* unit) that has not been assigned previously as an acronym has to be identified.
 1. If such a letter exists, this letter is assigned to the value (*id.* unit) as its acronym.

¹⁵⁹ This method could also be applied to the determination of the acronyms associated to the different attributes that can be ascribed to a particular (linguistic) unit. However, as shown below, attributes are not represented explicitly in this type of compressed tags. They are implicit, and values are related to their corresponding attribute by the position they occupy within the resulting (compressed) tag.

¹⁶⁰ This method for acronym choice was inspired by the method for assigning Chemical symbols to elements, which is somehow captured in the Periodical Table of Elements.

2. Or else, if no letters from the value (*id.* unit) English descriptor are available to designate it, then the first free letter from the alphabet is assigned to the value (*id.* unit) as its acronym.

Besides, OntoTag's annotation scheme also borrows from EAGLES (1996a) its **mechanism for the inclusion of underspecified or ambiguous morphosyntactic tags**, extending it to other levels of annotation. This mechanism permits assigning more than a value to a given attribute by putting the alternative values into brackets and separating them by means of vertical bars. For example, if it cannot be automatically discerned whether the Gender of an Adjective is Masculine (M) or Feminine (F) in a given context, then the value for this attribute is left underspecified in the output, *i.e.*, the attribute Gender is assigned an “[M|F]” value¹⁶¹. This mechanism is automatically enabled, in the case of unit underspecification, by means of the hierarchical organisation of ontologies¹⁶².

However, when determining **how the annotations of attributes and values would be structured**, it was decided that *the positional, implicit way to refer to attributes in EAGLES (1996a) recommendations* would not be included in OntoTag or implemented in OntoTagger, for the following reasons. Consider, for instance, the following POS tag, “**NCMSN**”, which is an EAGLES conformant tag that stands for a Noun (**N**) unit, which has been further subclassified into a Common Noun (**C**), whose Gender is Masculine (M), whose Number is Singular (S), and whose Case is Nominative (N). As can be observed, the positional format of EAGLES (1996a) assigns a particular order number to each attribute of the unit being annotated, and each attribute is assigned a fixed position in the string that represents its categorisation and its set of attribute values, according to this order number. Once this position has been assigned and fixed, the denomination of the attribute to which each value corresponds is implicit. Indeed, it can be unambiguously determined by its relative position *a priori* and, hence, it can be removed from the annotation. Now, consider this other EAGLES-conformant POS tag, “**NUCMSN**” (which stands for a Numeral (**NU**) that can be further subclassified into a Cardinal Numeral (**C**), whose Gender is Masculine (M), whose Number is Singular (S), and whose Case is Nominative (N)). When comparing these two POS tags (“**NCMSN**” and “**NUCMSN**”) no big difference can be observed between them. In fact, they can be easily mistaken when reviewing an annotated document. Hence, the (human-)readability of this type of annotations is jeopardised. In addition, certain units require an arbitrary number of characters for their representation (according to the degree of granularity of their subclassification). In the first example, “**NC**”, only two characters are required to represent the (sub)categorisation of the unit, whereas in the second example –“**NUC**”– three characters are required. This second aspect complicates

¹⁶¹ Note that this tag is not information-less: at least the Neuter (N) value has been eliminated from the possible ones.

¹⁶² In effect, when it cannot be automatically disambiguated which particular subtype of unit is being handled, it can be annotated by means of its parent unit in the LUO.

unnecessarily the automatic processing of the tags included in annotated documents. Consequently, it was decided to design an explicit, non-positional format of annotation, but keeping the compression of the values. This format of value annotation, together with the underspecification mechanism, also allowed for a virtual treatment of values as sets (instead of individuals) being assigned to attributes. The resulting annotation format proved itself to be very useful in the combination sub-phase of OntoTagger (see Section 5.3 for details).

Nevertheless, simple apposition of acronyms was allowed in OntoTag's annotation scheme and in OntoTagger's XML Schema for the **annotation of units**. Accordingly, units are subspecified by recursively appending, to the full acronym of its parent in the LUO (one of OntoTag's ontologies), the specific acronym of the corresponding subcategory, which differentiates it from its siblings. For example, the acronym for a `Countable Noun` is obtained by appending its specific acronym (C, as opposed to M, the specific acronym for a `Mass Noun`) to the acronym of its parent in the LUO. The parent of `Countable Noun` in the LUO is `Common Noun`, whose acronym is built by appending its specific acronym (C, as opposed to P, the specific acronym for a `Proper Noun`) to the acronym of `Noun` (N), which is its parent in the LUO. Therefore, the acronym for `Common Noun` is NC and, hence, the acronym for `Countable Noun` is, eventually, NCC.

The particular units, attributes and values that constitute the annotation triples in OntoTag's annotation scheme, together with the acronyms used to refer to each element (where appropriate), are explained below. Each `Linguistic Level` contemplated in this scheme is presented in a dedicated subsection.

4.3.1.1 MORPHOSYNTACTIC ANNOTATIONS

Regarding the annotation of the **units at the Morphosyntactic Level**, the major categories (Noun, Verb, Adjective, Adverb, etc.) determined in EAGLES (1996a) were further subcategorised in the development of the Linguistic Unit Ontology (LUO) of OntoTag. This was done using the `Type` attributes described in the aforementioned document, as well as some of the `Function` attributes for some of the categories, and also the attribute `Status` in the case of verbs. The remaining attributes were considered the proper attributes of the units and were included in OntoTag's Linguistic Attribute Ontology (LAO) and treated as such within OntoTag's annotation scheme. The result of this re-structuring of EAGLES (1996a) attributes originated the hierarchy of units shown in Table 89. In this table, the information related to each unit has been split into a pair of columns (Name, Acronym), and the children of a unit are represented on the right of their parent in the hierarchy (the unit in question). The differences between the units considered here and the units

directly derived from EAGLES (1996a) are discussed in detail in Subsection 4.1.3, where the LUO is presented.

Table 89: Morphosyntactic units in OntoTag(ger)'s annotation scheme(a)

Name	Acronym	Name	Acronym	Name	Acronym	Name	Acronym	
Adjective	AJ	Cardinal Adjective (1)	AJC NUCA					
		Ordinal Adjective (2)	AJO NUOA					
Adposition	AP	Circumposition	APC					
		Fused Prep-At	APF					
		Postposition	APP					
		Preposition	APR					
Article	AT[D I]	Definite Article	ATD					
		Indefinite Article	ATI					
		Partitive Article	ATP					
Adverb	AV	Degree Adverb	AVD					
		Particle Adverb	AVP					
		General Adverb	AVG	Non Wh- Adverb (3)	AVGN AV[G R]N AVRN			
		Pronominal Adverb	AVR	Wh- Adverb (3)	AVGW AV[G R]W AVRW	Exclamatory Adverb	AVGWE AV[G R]WE AVRWE	
						Interrogative Adverb	AVGWI AV[G R]WI AVRWI	
				Relative Adverb	AVGWR AV[G R]WR AVRWR			
Conjunction	C[C S]	Coordinating Conjunction	CC	Correlative CC	CCC			
				Initial CC	CCI			
				Non-Initial CC	CCN			
				Simple CC	CCS	Adversative Conjunction	CCSA	
		Copulative Conjunction	CCSC					
		Disjunctive Conjunction	CCSD					
		Subordinating Conjunction	CS	Adverbial SC	CSA			
				Comparative SC	CSC			
				With Finite SC	CSF			
				With Infinitive SC	CSI			
Substantive SC	CSS							
Interjection	I							
Noun	N	Common Noun	NC	Countable Noun	NCC			
				Mass Noun	NCM			
		Proper Noun	NP					
Numeral	NU[C O]	Cardinal	NUC	Cardinal Pronoun/Determiner (4)	NUC NUC[P D] PDNC PDNC[P D]	Cardinal Determiner	NUCD PDNCD	
						Cardinal Pronoun	NUCP PDNCP	

Name	Acronym	Name	Acronym	Name	Acronym	Name	Acronym
				Cardinal Adjective (1)	NUCA AJC		
		Ordinal	NUO	Ordinal Pronoun/Determiner (4)	NUO NUO[P D] PDNO PDNO[P D]	Ordinal Determiner	NUOD PDNOD
					Ordinal Adjective (2)	NUOA AJO	
Pronoun/Determiner	PD	Demonstrative Pronoun/Determiner (4)	PDD[P D] PDD	Demonstrative Determiner	PDDD		
				Demonstrative Pronoun	PDDP		
		Exclamatory Pronoun	PDEP				
		Indefinite Pronoun/Determiner (4)	PDI[P D] PDI	Indefinite Determiner	PDID		
				Indefinite Pronoun	PDIP		
		Interrogative Pronoun/Determiner (4)	PDT[P D] PDT	Interrogative Determiner	PDTD		
				Interrogative Pronoun	PDTP		
		Numeral Pronoun/Determiner (4)	PDN[P D] PDN	Cardinal Pronoun/Determiner (4)	PDNC PDNC[P D] NUC NUC[P D]	Cardinal Determiner	PDNCD NUCD
					Cardinal Pronoun	PDNCP NUCP	
		Ordinal Pronoun/Determiner (4)	PDNO PDNO[P D] NUO NUO[P D]	Ordinal Determiner	PDNO PDNO[P D] NUO NUO[P D]	Ordinal Determiner	PDNOD NUOD
					Ordinal Pronoun	PDNOP NUOP	
		Partitive Determiner	PDPD				
		Personal Pronoun	PDSP				
		Possessive Pronoun/Determiner (4)	PDO[P D] PDO	Possessive Determiner	PDOD		
				Possessive Pronoun	PDOP		
		Reciprocal Pronoun	PDRP				
		Reflexive Pronoun	PDFP				
		Relative Pronoun/Determiner (4)	PDL[P D] PDL	Relative Determiner	PDLD		
				Relative Pronoun	PDLP		
		Punctuation Mark	PU	Period	PU01		
Comma	PU02						
Question Mark	PU03						
Exclamatory Mark	PU04						
Colon	PU05						
Semi-Colon	PU06						
Question Mark (Open)	PU07						
Exclamatory Mark (Open)	PU08						
Simple Quotation Mark	PU[09 10] (5)			Simple Quotation Mark (Open)	PU09		
				Simple Quotation Mark (Close)	PU10		
Double Quotation	PU[11 12]			Double Quotation	PU11		

Name	Acronym	Name	Acronym	Name	Acronym	Name	Acronym		
		Mark	(5)	Mark (Open)					
				Double Quotation Mark (Close)	PU12				
		Apostrophe	PU13						
		Parenthesis	PU[14 15] (5)	Parenthesis (Open)	PU14				
				Parenthesis (Close)	PU15				
		Hyphen	PU16						
		Dash	PU17						
		Slash	PU18						
		Bracket	PU[19 20] (5)	Bracket (Open)	PU19				
				Bracket (Close)	PU20				
		Backslash	PU21						
Residual	R	Abbreviation	RA						
		Acronym	RC						
		Foreign Word	RF						
		Formula	RO						
		Symbol	RS						
		Unclassified	RU						
Unique/Unassigned	U								
Verb	V	Auxiliary	VA	Modal Auxiliary	VAM				
				Primary Auxiliary	VAP				
		Main	VM	Copular	VMC				
				Predicative	VMP				
		Semi-Auxiliary	VS	Aspectual Semi-Auxiliary	VSA				
				Modal Semi-Auxiliary	VSM				

Some comments must be made about the units in Table 89:

1. Cardinal Adjective is a subclass of both Adjective and Cardinal (Numeral) and has, therefore, two different acronyms (AJC and NUCA), depending on which of these major categories is taken as the main one describing the instance in question.
2. Analogously, Ordinal Adjective is a subclass of both Adjective and Ordinal (Numeral) and has, therefore, two different acronyms (AJO and NUOA) as well.
3. A Non Wh- Adverb and a Wh-Adverb can be either a General Adverb or a Pronominal Adverb. Therefore, these two subcategorisations (Non Wh- Adverb vs. Wh-Adverb and General Adverb vs. Pronominal Adverb) are rather independent from each other. The order in which each of these subcategorisations is performed and included in the acronyms has been determined according to the EAGLES (1996a) recommendations. However, both of them are sometimes difficult to disambiguate (automatically) and both of them might need to be underspecified. Hence, these acronyms had to represent this circumstance accordingly. This was achieved by allowing also the underspecification of the higher-level subcategorisation (*i.e.*, General Adverb vs. Pronominal Adverb). That is why they have been included here this way.

4. EAGLES (1996a) established a common major category for pronouns and determiners. This, in fact, is an underspecification of both categories, which might be justified by the difficulty to distinguish between them automatically in certain languages (such as English). Therefore, the distinction between them both is deferred as much as possible in the acronym of the unit, in order for this underspecification to be more easily applied. In these cases, two alternative acronyms have been allowed: the one making explicit the underspecification of these two categories, and the one leaving this information implicit.
5. For those subcategories of `Punctuation` not contemplated in EAGLES (1996a), an intermediate (extra) hierarchical level was allowed in order to represent these types of morphosyntactic units better. It must be noted, though, that the acronyms of this intermediate level concepts are obtained by the underspecified aggregation of the acronyms of their particular children in the hierarchy.

The remaining attributes in EAGLES (1996a), except for the adjectival attributes `NP Function` and `Use`, do not subcategorise any type of unit, and are considered proper **morphosyntactic attributes** in OntoTag. These proper morphosyntactic attributes are shown in Figure 13. As far as the attributes `NP Function` and `Use` are concerned, they have been conveniently extended (also renamed, in the case of `NP Function`, which is referred to as `Phrase Function` in OntoTag) and moved to the syntactic and the semantic levels, respectively. This is where they obtain their real meaning, and where they can be associated to other suitable units.

The different **morphosyntactic values** that can be assigned to the attributes in Figure 13 are shown in Table 90. Table 90 also shows the correspondence between the units and the attributes of the `MorphoSyntactic Level` in OntoTag. First, attributes and values with a white background are recommended within EAGLES (1996a). Second, those with a yellow background are considered optional, but generic (affecting most of the languages studied in the recommendations). Third, those with an orange background are considered optional but language-specific. Finally, those with a red background are minimal additions (also optional and language-specific) required to extend EAGLES (1996a) to Eastern European (Slavic) languages.

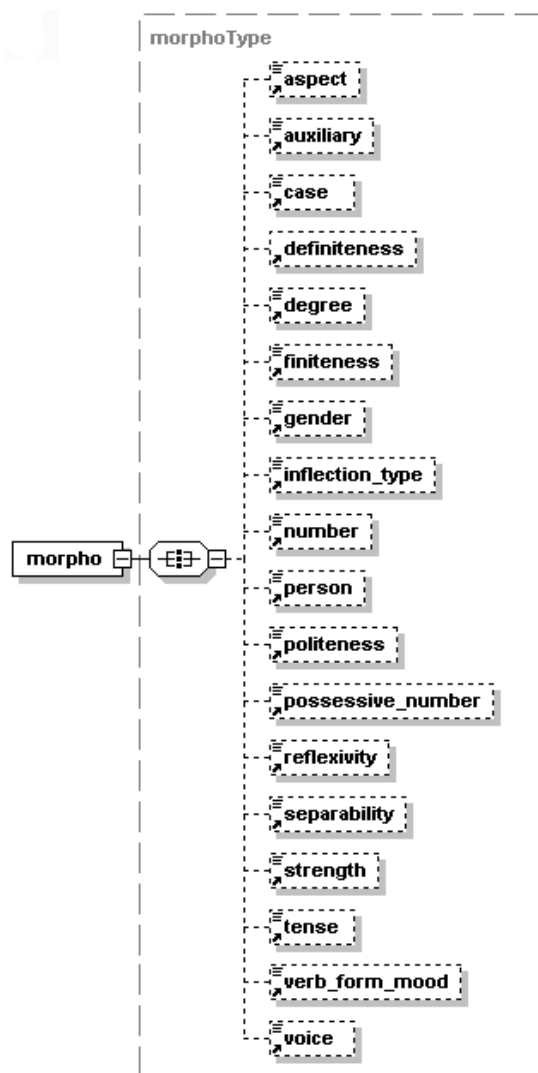


Figure 13: Morphosyntactic attributes in OntoTag(ger)'s annotation scheme(a) (XML Schema excerpt)

Table 90: Morphosyntactic values in OntoTag(ger)'s annotation scheme(a)

Unit(s)	Attribute	Value	Acronym
VERB	ASPECT	Imperfective	I
		Perfective	P
VERB	AUXILIARY	Be	B
		Have	H
NOUN	CASE	Ablative	A
ADJECTIVE		Accusative	C
		Dative	D
		Genitive	G
PRONOUN/DETERMINER		Indeclinable	I
ARTICLE		Instrumental	S
NUMERAL		Locative	L
NOUN		DEFINITENESS	Nominative
	Vocative		V
	Definite		D
ADJECTIVE	DEGREE	Indefinite	I
		Unmarked	U
		Comparative	C
ADVERB	FINITENESS	Positive	P
Superlative		S	
VERB	FINITENESS	Finite	F
		Non-finite	N

Unit(s)	Attribute	Value	Acronym
NOUN	GENDER	Common	C
VERB		Feminine	F
ADJECTIVE		Masculine	M
PRONOUN/DETERMINER		Neuter	N
ARTICLE			
NUMERAL			
RESIDUAL			
ADJECTIVE	INFLECTION_TYPE	Mixed	M
		Strong-Flection	S
		Weak-Flection	W
NOUN	NUMBER	Plural	P
VERB			
ADJECTIVE		Singular	S
PRONOUN/DETERMINER			
ARTICLE			
NUMERAL			
RESIDUAL			
VERB	PERSON	First	1
PRONOUN/DETERMINER		Second	2
		Third	3
PRONOUN	POLITENESS	Familiar	F
		Polite	P
PRONOUN/DETERMINER	POSSESSIVE_NUMBER	Plural	P
		Singular	S
VERB	REFLEXIVITY	Non-reflexive	N
		Reflexive	R
VERB	SEPARABILITY	Non-separable	N
		Separable	S
PRONOUN/DETERMINER	STRENGTH	Strong	S
		Weak	W
VERB	TENSE	Future	F
		Imperfect	I
		Past	P
		Present	R
VERB	VERB_FORM_MOOD	Conditional	C
		Gerund	G
		Imperative	I
		Indicative	N
		Infinitive	F
		-Ing form	O
		Participle	P
		Subjunctive	S
		Supine	U
VERB	VOICE	Active	A
		Passive	P

4.3.1.2 OTHER ANNOTATIONS AT THE SYNTACTIC LEVEL

Regarding the annotation of **units at the Syntactic Level**, the major categories of this level (namely Phrase, Clause and Sentence) were already included and subcategorised in EAGLES (1996b). Hence, they were incorporated as syntactic units into the LUO. However, another Syntactic Unit had to be added to these three, for the sake of completeness. This additional unit is referred to as Token and represents a Morpho-Syntactic Unit conveniently

subcategorised¹⁶³. A Token can be further subcategorised as a Simple Token or a MultiWord Token. Obviously, this subcategorisation is made according to the number of words that constitute each Token. For example, whereas ‘Inés Sastre’ is a MultiWord Token, ‘director’ is a Simple Token. The resulting hierarchy of syntactic units is shown in Table 91. As with Table 90, the information related to each Syntactic Unit has been split into a pair of columns (Name, Acronym), and the children of each Syntactic Unit are represented on the right of their parent in the hierarchy (that is, the Syntactic Unit in question).

Table 91: Syntactic units in OntoTag(ger)’s annotation scheme(a)

Name	Acronym	Name	Acronym
Clause	CL	Adverbial Clause	CL-ADV
		Comparative Clause	CL-COMP
		Nominal Clause	CL-NOM
		Relative Clause	CL-REL
Phrase	P	Adjective Phrase	ADJP
		Adverb Phrase	ADVP
		Noun Phrase	NP
		Prepositional Phrase	PP
		Verb Phrase	VP
Sentence	S	Complex Sentence	S-C
		Compound Sentence	S-O
		Simple Sentence	S-S
Token	T	MultiWord Token	TM
		Simple Token	TS

As for the syntactic attributes considered in EAGLES (1996b), they are too (H)PSG-oriented, and needed to be extended with some additional syntactic attributes coming from other grammatical theories. Fortunately, the kind of syntactic analysis included in OntoTagger was of a functional- and dependency-related nature (obtained by means of Connexor’s FDG Parser). This helped determine a complementary set of syntactic attributes to the ones included in the aforementioned recommendations. Therefore, the **syntactic attributes** (and, hence, also the values) included in OntoTag’s annotation scheme are the (H)PSG-oriented ones considered in EAGLES (1996b) plus the functional- and dependency-oriented ones annotated by Connexor’s FDG Parser. Apart from these, the Phrase Function attribute mentioned above was also included in this level of the annotation scheme. The Phrase Function attribute is a generalisation of a morphosyntactic attribute included in EAGLES (1996a), namely the *NP Function* attribute. This morphosyntactic attribute was ascribed

¹⁶³ This unit is already included in the ISO/MAF (2008) and ISO/SynAF (2010) standard drafts.

solely to adjectives there. On the contrary, its generalisation, the `Phrase Function` attribute, is a `Syntactic Attribute` that can be ascribed to any kind of `Token`¹⁶⁴.

The eventual syntactic attributes and relationships included in OntoTag's annotation scheme are shown in Figure 14.

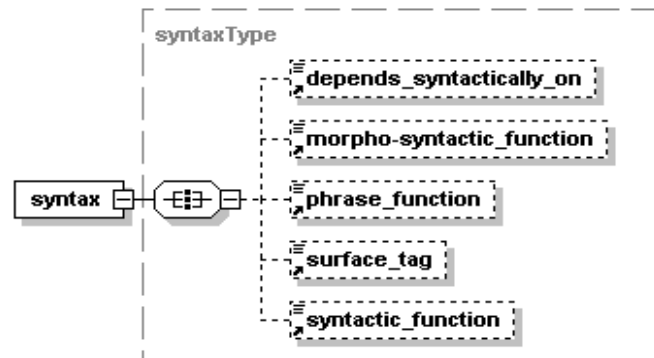


Figure 14: Syntactic attributes in OntoTag(ger)'s annotation scheme(a) (XML Schema excerpt)

The `depends_syntactically_on` element details the annotation of a **syntactic relationship** (a dependency relation, in fact). This annotation consists of (i) a pointer from the `Token` in question to another `Token` in the same sentence (the one on which the former `Token` depends on) and (ii) a label that identifies the particular type of dependency holding between them both. As for the pointer to the other `Token`, it was implemented by means of a particular identifier built out of (1) the number of order of the `Token` in question within its `Sentence`, (2) the number of order of this `Sentence` within its `Paragraph`, and (3) the number of order of this `Paragraph` within the input document – see Section 5.5.1 and Section 5.5.2 for details. As for the label identifying the type of dependency, each `depends_syntactically_on` element stands for a labelled edge in a dependency graph. Such a labelled edge represents the particular syntactic dependency existing between two tokens within an input sentence. The different labels included in OntoTag for tagging dependencies are shown in Table 92. Note that some of the tags in this table are of a syntactic nature (*e.g.*, `subj` – Subject) and some are of a semantic nature (*e.g.*, `sou` – Source). In fact, the latter ones were used in OntoTagger as a sort of role label for the `Token` in question, as values of the `Participant Type` semantic attribute (described in the following subsection).

The different **syntactic values** that can be assigned in OntoTag to the attributes in Figure 14 are shown in Table 93. This table shows also the correspondence between the units and the attributes of the `Syntactic Level`.

¹⁶⁴ For example, the `Prepositional Phrase` 'of Spain' has a `Postmodifying Phrase Function` in the `Noun Phrase` 'The King of Spain'.

Table 92: Connexor's FDG parser dependency functions

Tag	Explanation
main	main element
v-ch	verb chain
pm	prepositional marker
pcl	preverbal particle
phr	verb particle
subj	subject
obj	object
comp	subject complement
dat	indirect object
oc	object complement
copred	copredicative
voc	vocative
tmp	time
dur	duration
frq	frequency
qua	quantity
man	manner
loc	location
sou	source
goa	goal
ent	contingency (purpose or reason)
rsn	reason
pur	purpose
end	condition
com	comitative
meta	clause adverbial
cla	clause initial adverbial
ha	heuristic PP attachment
qn	quantifier
det	determiner
neg	negator
ada	attributive nominal
ads	adjectival postmodifier
mod	other postmodifiers
ad	attributive adverbial

Table 93: Syntactic values in OntoTag(ger)'s annotation scheme(a)

Unit(s)	Attribute	Value	Acronym
TOKEN	MORPHO-SYNTACTIC_FUNCTION	Adjective	AJ
		Adposition	AP
		Adverb	AV
		Article	AT
		Conjunction	C
		Interjection	I
		Noun	N
		Numeral	NU
		Pronoun/Determiner	PD
		Punctuation Mark	PU
		Residual	R
		Unique/Unassigned	U
		Verb	V
	PHRASE_FUNCTION	Head-function	H
		Postmodifying	P
		Premodifying	R
	SURFACE_TAG ¹⁶⁵	Adjectival	A
		Adverbial	V
		Coordinating Conjunctive	C
		Determiner	D
		Nominal	N
		Predicator	P
		Prepositional Marker	R
		Quantifier	Q
		Unspecified	U
	SYNTACTIC_FUNCTION	Adjunct	A
		Direct Object	D
		Disjunct	J
		Indirect Object	I
		Object Complement	O
		Predicator	P
		Predicator Complement	R
		Prepositional Object	E
Subject		S	
Subject Complement		U	

4.3.1.3 ANNOTATIONS AT THE SEMANTIC LEVEL

The Semantic Level annotations contemplated in the annotation scheme of OntoTag are described in this section. They are focused on the lexical semantic annotation of the tokens included in the input documents. As described previously, these semantic annotations are distributed in OntoTag between the Concept Semantic Annotation Layer and the Instance Semantic Annotation Layer. The annotations pertaining to the Concept Semantic Annotation Layer are presented in Subsection 4.3.1.3.1, and the ones relating to the Instance Semantic Annotation Layer are presented in Subsection 4.3.1.3.2.

¹⁶⁵ Included, for example, in Connexor's FDG parser annotations.

4.3.1.3.1 Annotations at the Concept Semantic Annotation Layer

Regarding the annotation of **units of the Concept Semantic Annotation Layer**, only the major subclasses of the `Semantic Unit` class in the LUO are used in OntoTag’s annotation scheme.

These major subclasses are shown in Table 94. The semantic units included in this table are used in this scheme as a kind of high-level semantic annotation. This is due to the fact that it is recommended to use (Euro)WordNet synsets or the concepts of a domain ontology (for example, the CNEO, in the case of OntoTagger) to characterise each of these units within the Concept Semantic Annotation Layer. This is attained by means of their `word_sense` attribute, described below.

Table 94: Semantic units from OntoTag’s LUO in OntoTag(ger)’s annotation scheme(a)

Name	Acronym
Circumstance	C
Entity	E
Process ¹⁶⁶	P
Property	R

The **semantic attributes and relationships of the Concept Semantic Annotation Layer** included in OntoTag’s annotation scheme are shown in Figure 15. First, as with the `Phrase Function` attribute, the `Use` attribute is a generalisation of the morphosyntactic attribute with the same name in EAGLES (1996a). It was assigned initially to adjectives in the recommendations, but it has been generalised here so that it can be attached to any `Property`. Second, the `Participant Type` attribute details the semantic role that a `Semantic Unit` plays in the `Sentence` where it appears. Therefore, the `Participant Type` attribute implements the Semantic Role Labelling Layer within OntoTag. Third, the `word_sense` element mentioned above is the `Semantic Attribute` responsible for the Concept Sense-Tagging Layer in OntoTag. Fourth, the `Synonymy`, `Holonymy`, `Hyperonymy` and `Meronymy` elements are used to annotate the corresponding lexical semantic relations holding for the `Semantic Unit` in question. Finally, there is an attribute of the element `word_sense` (referred to as `Domain`), not shown in Figure 15 (but shown in the examples of Section 5.5.1 and Section 5.5.2), responsible for the Semantic Field Tagging Layer in OntoTag.

¹⁶⁶ This is the preferred term in OntoTag and in OntoTagger to refer to an `Event` (a suitable synonym has been included in OntoTag’s ontologies).

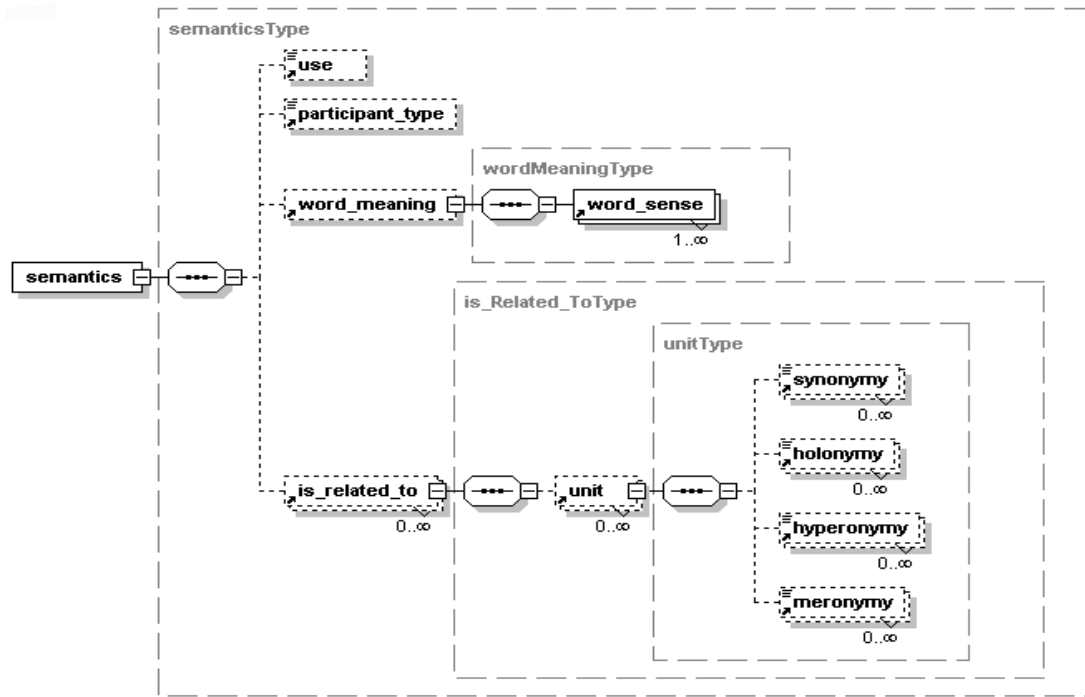


Figure 15: Semantic attributes and relations in OntoTag(ger)'s annotation scheme(a) (XML Schema excerpt)

The different **semantic values** that can be assigned in OntoTag to the semantic attributes and relations in Figure 15 are shown in Table 95.

Table 95: Semantic values in OntoTag(ger)'s annotation scheme(a)

Unit(s)	Attribute	Value	Acronym
PROPERTY	USE	Attributive	A
		Predicative	P
ENTITY	PARTICIPANT_TYPE	Actor	A
		Attribute	U
		Behaver	B
		Carrier	C
		Existent	E
		Goal	G
		Identified	I
		Identifier	D
		Phenomenon	P
		Recipient	R
		Sayer	S
		Senser	N
		Target	T
		Token	O
Value	V		

Table 95 also shows the correspondence between the units and the attributes of the Concept Semantic Annotation Layer. First, the values of the Use attribute are the ones included for it in EAGLES (1996a). Second, the values of the Participant Type attribute have been extracted from the Functional Grammar of Halliday (1994;1996). Third, the possible values recommended in OntoTag for the Word Sense attribute and the Synonymy, Holonymy, Hyperonymy and Meronymy relations are (i) the identifiers of (Euro)WordNet synsets and/or (ii) the names of the

concepts of a suitable domain ontology or an equivalent identifying attribute within that ontology. A concrete sense tagset developed according to this recommendation can be found in the description of OntoTagger's annotation schemas (Section 5.5). Lastly, as for the values of the `Domain` attribute, OntoTag's proposal is to extract them, for example, from the EuroWordNet Top Ontology included in SIMPLE (2000), which has been reused and conveniently linked to the LUO towards this aim.

4.3.1.3.2 Annotations at the Instance Semantic Annotation Layer

An initial remark has to be made about the way in which the Instance Semantic Annotation Layer was handled in OntoTag(ger)'s annotation scheme(a). Originally, this layer (dealing mainly with the annotation of named entities) was judged to stand at the `Discourse Level`. This was due to the fact that they are the source of several discourse relationships (such as an anaphora) or the target of some others (such as a cataphora) and they establish in many cases a particular kind of linking between text and reality. Hence, from this point of view, named entities are units of the `Discourse Level`, since they contribute substantially to discourse coherence. Nevertheless, named entities have a particular projection on the `Semantic Level`, since their sense (or meaning) must be annotated as well¹⁶⁷.

Consequently, named entities can be said to belong to both the semantic and the discourse levels. In fact, named entities stand in the interface between them both. Therefore, a hybrid annotation of them had to be allowed in OntoTag's annotation scheme. This was achieved as follows. First, named entities must be annotated as instances of the `Named Entity` unit of the LUO. In this ontology, `Named Entity` is a *Subclass* of both `Entity` (which, in turn, is a *Subclass* of `Semantic Unit`) and `Discourse Unit`. Therefore, named entities can be (i) included in the `Discourse Level` for their posterior annotation as sources or targets of discourse relations and (ii) annotated as semantic units with a particular sense tag within the Instance Semantic Annotation Layer.

For this reason, and from a theoretical point of view, the annotations of this layer are ontology instances that belong to (at least) two classes: (i) `Named Entity` (included in the LUO and imported in the CNEO) or any of its top-level subclasses in the CNEO and (ii) the corresponding class (or concept) of which they are members (included in a domain ontology). Therefore, the **units included in the Instance Semantic Annotation Layer** of OntoTag's annotation scheme are exclusively `Named Entity` and its top-level subclasses in the CNEO. This top-level subclassification of `Named`

¹⁶⁷ However, unlike other semantic units, their sense cannot be represented by the classes included in the semantic module of the LUO. This is due to the fact that named entities refer, in general, to instances of concepts (that is, individuals), instead of concepts (that is, classes of individuals). For example, the `Named Entity` 'Juan Carlos I' can be regarded as an instance of the concept `King`.

Entity is shown in Table 96. Once again, the information related to each unit (or concept) has been split into a pair of columns (Name, Identifier). As usual, the children of a unit are represented on the right of their parent in the hierarchy (the unit in question).

Table 96: Concepts from the CNEO used for named entity annotation in OntoTag's annotation scheme

Name	Identifier	Name	Identifier	Name	Identifier	Name	Identifier		
Named Entity	1	Proper Named Entity	10	Animated Entity	100	Human Entity	1000		
						Non Human Entity	1001		
				Non Animated Entity	101	Organization Entity	1010		
						Geographical Location Entity	1011		
						Geopolitical-Historical Location Entity	1012		
						Domain Entity	1013		
				Number Expression	11	Ratio	110	Percent	1100
						Absolute Quantity	111	Money	1110
								Duration	1111
				Age	1112				
		Temporal Expression	12	Time	120				
				Date	121	Month	1210		
						Century	1211		
						Day	1212		
						Year	1213		

As can be observed in Table 96, this unit subclassification is based on the MUC-7 and ACE *de facto* standards for Named Entity recognition and subclassification. Accordingly, it is most suitable to describe the units annotated within the Instance Semantic Annotation Layer. As a matter of fact, the input tokens annotated by means of any of the units in this tagset are included by OntoTagger as instances of their corresponding concept within the CNEO.

The subclassification of each concrete Named Entity is detailed in OntoTag's annotation scheme by means of two dedicated **instance semantic attributes**, namely MUC-7 Tag and Subcategory. On the one hand, MUC-7 Tag characterises a Named Entity according to the MUC-7 *de facto* standard. The annotation of this attribute is considered recommended for this level in OntoTag. On the other hand, Subcategory characterises in more detail the sense of the Named Entity, according to a more specific ontology (usually a domain one). This attribute constitutes an optional type of annotation of this layer in OntoTag.

The **semantic values** that each of these two attributes can take are the strings corresponding to the identifier (or else, the name) of the class that characterises the Named Entity in question. This class can be part of the LUO, the CNEO or any other (domain) ontology.

5. ONTOTAGGER: AN INSTANCE OF ONTOTAG

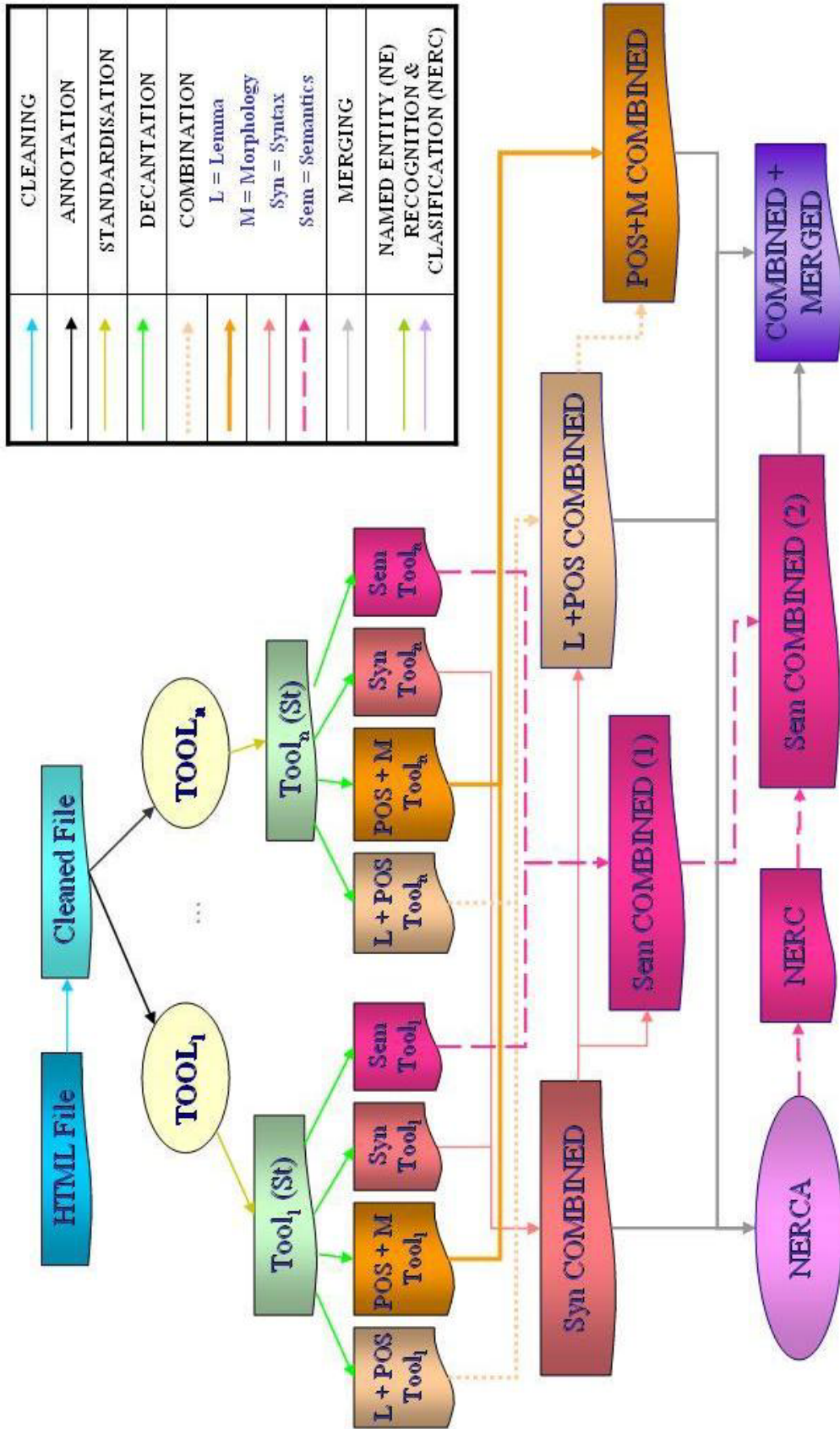
In order to assess and evaluate the OntoTag model (Aguado *et al.*, 2003), a first prototype, OntoTagger, was implemented for Spanish, in the domain of entertainment (focused mainly on the area of film reviews). The particular configuration of this first implementation is shown in Figure 16. This section is devoted to the description of OntoTagger, that is, (i) the components of this architecture configuration, which is an instance of OntoTag’s (abstract) annotation architecture, focusing on the key points of their design and implementation; and (ii) its associated annotated schemas, which, in turn, implement OntoTag’s (abstract) annotation scheme. First of all, the different linguistic annotation tools integrated into OntoTagger will be presented in Subsection 5.1. Second, the main components of OntoTagger’s configuration will be introduced in Subsection 5.2. Third, main modules of this configuration, namely the combination and the semantic annotation modules, will be described in detail (in subsections 5.3 and 5.4, respectively). Finally, the schemas implementing OntoTag’s annotation scheme in OntoTagger will be shown in Subsection 5.5.

5.1. LINGUISTIC ANNOTATION TOOLS INTEGRATED INTO ONTOTAGGER

The different linguistic annotation tools integrated into OntoTagger are described in this subsection, before introducing the main components (or modules) of OntoTagger. As has already been stated, the main goal of OntoTagger is to combine and integrate the annotations of the different (linguistic) annotation tools included in its concrete configuration. As explained in Chapter 2, where the goals of OntoTag were presented, the tools incorporated into OntoTagger had been chosen beforehand¹⁶⁸.

Hence, the linguistic tools finally integrated in OntoTagger where: (i) Connexor’s FDG (Functional Dependency Grammar) Parser 3.7 for Spanish, (ii) Bitext’s DataLexica and (iii) a POS tagger developed at Universidad de Murcia (also referred to as LACELL’s POS tagger here). These three tools as well as the reason for choosing them are described in detail in the following subsections.

¹⁶⁸ They had been purchased for the research group in which this research was carried out.



CLEANING	↑
ANNOTATION	→
STANDARDISATION	↑
DECANTATION	↑
COMBINATION	↑
L = Lemma	↑
M = Morphology	↑
Syn = Syntax	↑
Sem = Semantics	↑
MERGING	↑
NAMED ENTITY (NE) RECOGNITION & CLASSIFICATION (NERC)	↑

Figure 16: OntoTagger's configuration

5.1.1. CONNEXOR'S FDG (FUNCTIONAL DEPENDENCY GRAMMAR) PARSER FOR SPANISH

This tool¹⁶⁹ performs Lemma and POS tagging, as well as a fashion of dependency parsing, which gives information about word-to-word syntactic dependency relations and also about other syntactic attributes based on a linguistic formalism known as Functional Dependency Grammar (Tapanainen & Järvinen, 1997). According to the user's manual of this tool, its lexicon contains about 40,000 lexical entries, arranged into inflectional and derivative classes, and a heuristic guesser that helps tag unrecognised words. Its lexical entries have been semi-automatically extracted from texts and word-lists. In addition, some of them have been checked manually, according to Spanish grammars and dictionaries.

Figure 17 shows an example of the annotations performed by this tool, namely the dependency parsing tree for the Spanish sentence “*CC. OO. denuncia la grave situación económica de la diputación provincial.*”¹⁷⁰. As shown in this example, dependency relations are arranged in a way that permits representing an FDG (dependency) parsing as a tree. In such a tree, (a) the nodes are the tokens of the sentence (not only the leaves, as with (H)PSG parse trees); and (b) the edges are labelled with the particular type of dependency that holds between the nodes in question¹⁷¹.

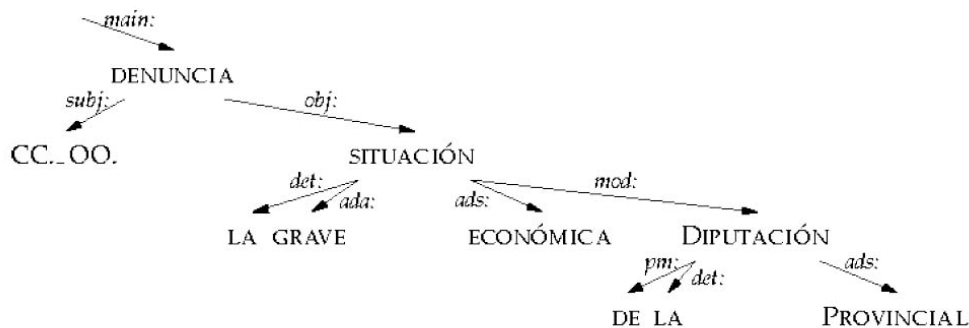


Figure 17: A dependency parsing tree of FDG (extracted from its user's manual)

The annotation of this example sentence performed by the tool has also been summarised in Table 97. As shown in this table, the output of FDG¹⁷² consists of: (a) the number of order of the Token tagged (its index); (b) the Word Form of the Token, that is, the string that appears in the input text;

¹⁶⁹ More accurately, the release integrated was 3.7.

¹⁷⁰ “CC. OO. [the acronym for ‘Comisiones Obreras’, a Spanish trade union] denounces the serious financial position of the provincial delegation.”

¹⁷¹ The example in Figure 17 and other examples of dependency parsing can be found in the user's manuals of the tool (or at Connexor's homepage¹⁷¹), as well as a detailed explanation of the features of FDG, including the exact meaning of each and every label used in its annotations.

¹⁷² The acronym FDG is use henceforth to refer to Connexor's FDG Parser.

(c) its Lemma; (d) its Dependency Tag, which specifies both the index of the Word or Token on which it depends and the type of syntactic relation holding between them; (e) its Surface Syntactic Tag, which basically details the syntactic function of the Word or Token in its Phrase (it can be a Head, a Premodifier or a Postmodifier); and (f) its morphosyntactic tags (*i.e.*, its grammatical category and its morphological attributes).

Table 97: FDG annotation example

INDEX	WORD FORM	LEMMA	DEPENDENCY TAG	SURFACE SYNTACTIC TAG	MORPHOSYNTACTIC TAGS
1	CC. OO.	cc. oo.	subj:>2 [subject]	&NH [nominal head]	<Abbr> N SG [abbreviated common noun, singular]
2	denuncia	denunciar	main:>0 [main predicate]	&+FM [finite main verb]	V IND PRES SG3 [verb, indicative, present, 3 rd person, singular]
3	la	la	det:>5 [determiner]	&DN> [determiner]	DET FEM SG [determiner, feminine, singular]
4	grave	grave	ada:>5 [attributive adjective]	&A> [nominal pre-modifier]	A COM SG [adjective, gender undetermined, singular]
5	situación	situación	obj:>2 [object]	&NH [nominal head]	N FEM SG [common noun, feminine, singular]
6	económica	económico	ads:>5 [post-modifying adjective]	&<A [nominal post-modifier]	A FEM SG [adjective, feminine, singular]
7	de	de	pm:>9 [preposition]	&PM> [preposition marker]	PREP [preposition]
8	la	la	det:>9 [determiner]	&DN> [determiner]	DET FEM SG [determiner, feminine, singular]
9	diputación	diputación	mod:>5 [modifier]	&NH [nominal head]	N FEM SG [common noun, feminine, singular]
10	provincial	provincial	ads:>9 [adjectival post-modifier]	&<A [nominal post-modifier]	A COM SG [adjective, gender undetermined, singular]
11	.	.			

For instance, in this example sentence, “*CC. OO.*” is tagged with the following information: (a) it is the *first* Token in the Sentence; (b) its Word Form is “*CC. OO.*”; (c) its Lemma is “*cc. oo.*”; (d) it is the Subject (*subj*) of the Token referenced by the index “2”; (e) it is the Head (“&NH”) of the Nominal Group it is part of; and (f) its morphological annotation states that it is an Acronym or an Abbreviation (<Abbr>) and also that it is a Common Noun (“N”) with a Singular Number (“SG”).

An important **drawback** of this tool is the degree of **ambiguity** that it introduces in its annotations (for example, Table 98 shows how FDG annotates the word “como” in the Spanish Sentence “Algunos de los personajes desaparecen sin saber como ni porqué” ≡ ‘Some of the characters disappear, nobody knows how or why’). This ambiguity is expressed by the appearance of multiple labels related to the same phenomenon being tagged for a large number of the tokens annotated. This degree of ambiguity affects both the annotation of the morphosyntactic category of some tokens and, in certain occasions, of some morphological attributes. However, the degree of annotation ambiguity introduced by this tool in the former case is certainly not as marked as in other tools, such as the next one described here, DataLexica.

**Table 98: FDG annotation of “como” in the Spanish Sentence
“Algunos de los personajes desaparecen sin saber como ni porqué”
(≡ ‘Some of the characters disappear, nobody knows how or why’)**

INDEX	WORD FORM	LEMMA	DEPENDENCY TAG	SURFACE SYNTACTIC TAG	MORPHOSYNTACTIC TAGS
11	como	como	&PM>	PREP	-
			&PM>	CS	-

According to the user’s manual, no more than 3 percent of the tokens are expected to be ambiguously POS-tagged; supposedly, this figure includes the degree of ambiguity on morphological attribute tagging, which empirically has proven to be much higher (specially gender tagging for adjectives). After manually-checking its annotations on a sample corpus (see Chapter 6 –Results and Evaluation), it was seen that around 19 percent of the tokens had an ambiguous grammatical category tag (that is, more than one category tag for a given token), and that approximately 14 percent of the overall morphosyntactic attributes had been ambiguously annotated as well. This degree of ambiguity reduces to some extent the reliability of its annotations.

The main **contribution** of this tool to OntoTagger’s overall performance are (a) the annotation of syntactic dependencies (*i.e.*, a fashion of syntactic parsing) and other syntactic elements, such as the syntactic surface tags, which are useful in the processing of verbal chains (complex and verbal forms and phrases, etc.); and (b) its reliability at annotating infrequent tokens (supposedly performed by its heuristic guesser, using their contextual information).

5.1.2. BITEXT’S DATALEXICA

This linguistic tool is described as a ‘lexical database that includes a complete encoding of the linguistic properties of words’¹⁷³. In practice, it behaves as a non-disambiguating Lemma and POS

¹⁷³ <http://www.bitext.com/ES/datalexica.asp>.

tagger. It was first developed for Spanish and then extended to cover other languages, such as Catalan or English. Its lexicon contains a large number of Spanish words (in the order of 2,000,000 words). This makes of DataLexica an appropriate and reliable tool for the morphological annotation of Spanish texts, provided that it is used in conjunction with a morphosyntactic disambiguating tool. Following the user’s manual of the tool, the entries of its lexicon are stored according to two different formats: an inflective format, for verbal forms, and a derivational format, for augmentatives, diminutives, comparative and superlative adjectives, etc. This lexicon is kept updated by means of a continued inspection of representative text databases of current Spanish.

The ambiguous annotation of the Spanish word “*como*” (≈ “(I) eat” / “as” / “like”) outputted by DataLexica is shown in Table 99.

Table 99: DataLexica annotation example: tags for the Spanish word “*como*”

Text	Lemma	Root (R) / Form (F)	POS	Inflectional attributes	Derivational attributes
como	comer	F	4	27	0
como	comerse	F	4	27	0
como	como	R	3	00	0
como	como	R	8	00	0

As can be observed in Table 99, DataLexica returns the annotation of each token split into five fields: (a) its Lemma, (b) a kind of Boolean value indicating whether it is an Inflected Form (“F”) or not (“R”), (c) a number representing its morphosyntactic category, (d) a code which stands for the inflectional information of the Token, and (e) another code denoting its derivational properties, such as the Degree in adjectives (Positive, Comparative or Superlative) or other word-formation features – for example, the addition of a ‘-mente’ Suffix (≈ ‘-ly’, in English) to an adjectival Root in order to obtain an Adverb.

The meaning of all the codes shown in the example above for morphosyntactic categories, and also of the inflective and derivational attributes, can be found in the tool’s user’s manual. As for the aforementioned example, Table 99 shows that several annotations can be obtained for a given token. In the case of “*como*”, the first two annotations stand for a Verb (POS = 4), in Present Tense, Indicative Mood, 3rd Person, Singular Number (inflectional attribute = 27)¹⁷⁴; the third one stands for an Adverb (POS = 3); and the last one stands for a Conjunction (POS = 8). It must be remarked that, on the one hand, none of these results is wrong for the word in question out of a particular context; but, on the other hand, it gives no hint on which can be its particular tag in the context where it appears.

¹⁷⁴ Each verb with a reflexive sense is double-annotated by DataLexica with a reflexive and a non-reflexive sense (which might be arguable).

This is a most prominent **drawback** of this tool: it is highly **ambiguous** at morphosyntactic annotation. In fact, its annotations lack any kind of disambiguation. This lack of disambiguation is expressed also in this case by the appearance of multiple labels associated to a large number of the tokens annotated (see Table 99). Besides, as OntoTagger was being developed, it was found that, despite of the indications of this tool user’s manual, the tool failed (frequently) on the annotation of augmentatives and diminutives.

The main **contributions** of this tool to OntoTagger’s overall performance are (a) the accuracy of its annotations, once the right one has been discerned from the spurious ones; and (ii) its robustness. In fact, it annotates almost all the tokens in any document (however, it does not annotate punctuation marks, for example). These two advantages can be attributed to the huge dimension of its lexicon.

5.1.3. LACELL’S POS TAGGER

This tool was developed within the LACELL (*Lingüística Aplicada Computacional, Enseñanza de Lenguas y Lexicografía*¹⁷⁵) research group, headed by Prof. Aquilino Sánchez, from Universidad de Murcia. LACELL’s POS tagger operates mainly at the MorphoSyntactic Level (including its Lemma Tagging Layer) and, in addition, it provides some particular semantic information. For example, this tool classifies and annotates adverbs according to a traditional sub-classification in Spanish grammar, based on their Semantic Function (or Role) in a Sentence, *i.e.*, as Manner, Location, Time, etc.

LACELL’s POS tagger was specifically developed for Spanish. Due to this fact, it can recognise and tag a large number of input tokens. The output of this tool for a Token contains four main fields: (a) the Word Form, as it appears in the input text; (b) the Lemma associated to the Word Form in question; (c) a numerical code representing its morphosyntactic category; and (d) another numerical code which stands for the value of each and every morphological attribute corresponding to its morphosyntactic category. An annotation example (the annotation of the Spanish word “*como*” in the context: “...*este deporte está hecho para gente como él.*”¹⁷⁶) is shown in Table 100.

Table 100: LACELL’s POS tagger annotation example

Text (Word Form)	Lemma	POS	Inflectional attributes
como	como	162	100

As can be observed in Table 100, this occurrence of “*como*” (whose Lemma is “*como*” as well) functions as an Adverbial Subordinating Conjunction (POS = 162) with no particular

¹⁷⁵ Applied and Computational Linguistics to Language Teaching and Lexicography.

¹⁷⁶ “...this sport was made for people of his kind”

inflective attributes (inflectional attributes = 100) in the context where it appears. The accuracy of the POS tag for this Token contrasts significantly with the tags attached to it by the rest of the tools, which merely stated that it is a `Conjunction` (if correctly tagged).

The most prominent **drawback** of this tool is that, for several input tokens, it does not annotate the value of their morphological attributes. In addition, it does not recognise the ends of paragraph (whilst the other two tools do).

The main **contribution** of LACELL's POS tagger to OntoTagger's overall performance is that the morphosyntactic annotation of this tool for a token is never ambiguous. That is, though it can be wrong, it just returns one POS tag for each input Token. However, its error rate in this aspect is rather low). Besides, it tags morphosyntactic categories according to a more fine-grained taxonomy (including a semantic classification for adverbs) than the ones of the other tools, that is, it is more precise.

After introducing the different linguistic annotation tools integrated into OntoTagger, we describe the main components of this OntoTag's configuration. This is done in the following subsection.

5.2. ONTOTAGGER'S CONFIGURATION COMPONENTS

When developing the prototype OntoTagger, several modules had to be built in order to enable the prototype, for example, to handle other external components, such as (1) OntoTag's ontologies, which provide the knowledge for the standardisation, comparison, combination and integration of annotations, and (2) the linguistic annotation tools, which provide the raw annotation data to be standardised, compared, combined and integrated. Some other modules had to be programmed and incorporated into this configuration as well. They perform other types of annotations not offered by the linguistic annotation tools, or else implement the configuration itself. Here is a description of the most prominent and interesting modules included in the development of OntoTagger:

- **Ontology Manager Module (OMM):** This module of OntoTagger deals with OntoTag's (linguistic) ontologies. The terms in these ontologies (concepts, attributes, values, etc.) are used in the annotation of documents to standardise and enable the comparison of the annotations produced by different linguistic annotation tools. Since OntoTag's ontologies are stored in WebODE¹⁷⁷, the OMM has to establish the communication between OntoTagger and WebODE, thus allowing OntoTagger to retrieve the content of these ontologies. Every time OntoTagger is run, a

¹⁷⁷ WebODE (Web Ontology Development Environment, <http://webode.dia.fi.upm.es/webode/login.html>) is a platform for the development of ontologies, built within the Ontology Engineering Group (OEG - <http://www.oeg-upm.net>) from the UPM (Universidad Politécnic de Madrid).

connection with WebODE is established and the state of each of the ontologies registered in OntoTagger is checked. This module also keeps (1) a list of the ontologies available in WebODE that have been registered for its use within OntoTagger; and (2) a list of the ontologies enabled for annotation at a given moment.

- **Tool Manager Module (TMM):** This module is in charge of (1) the addition and the elimination of (external) linguistic annotation tools from the system; and (2) updating the information related to these tools. Additionally, the TMM is responsible for (a) executing each of the annotation tools integrated in OntoTagger on each input document; and (b) collecting their output annotations. OntoTagger also keeps track of (i) the different annotation tools already registered in the prototype for annotation; (ii) the registered annotation tools that have been activated by the user at a given time; and (iii) the different libraries required for the execution of the linguistic annotation tools on a document. Keeping track of this information increases the modularity of the configuration and its independence from the tools incorporated in the process of annotation.

Moreover, the libraries mentioned in (iii) deal with the standardisation of the results of the linguistic annotation tools. In general, these results (*i.e.*, annotations) are usually obtained according to some particular annotation commitments or policies, even though a given (common) tagset and formalism might be used or followed. These annotation commitments or policies prevent the annotations of different tools from being compared and combined. This problem was solved within OntoTag and its configurations by means of the inclusion of a standardisation phase, which homogenises the annotations and enables their comparison and combination.

The following examples, found while implementing OntoTagger, show the effects of these different annotation commitments and policies on the annotations outputted by the linguistic tools incorporated into the configuration:

1. The correspondence between the tagset of a given linguistic tool and the terms conceptualised in OntoTag's ontologies is not a mathematical application in any of its two senses. On the one hand, a particular input tag might need to be mapped onto two or more term codes (not ambiguous) for its standardisation. For example, all the inflectional attributes associated to a Token are compressed into just one tag in DataLexica and in LACELL's tagger (unlike in Connexor's FDG Parser). Then, they have to be extracted and separated before they can be compared and combined. On the other hand, annotating a single phenomenon in a standard way sometimes requires processing two or more input tags at a time (as, for example, when annotating multiword tokens and named entities).

2. Clitic pronouns¹⁷⁸, for example, are annotated differently by each tool (if annotated at all). Accordingly, their annotation must be uniformed in the standardisation phase, since this affects the token alignment process, as well as the semantic tagging of some verbal forms in Spanish (as, for instance, the occurrences of verbs which can be reflexive or not)¹⁷⁹.

As mentioned above, this module is in charge of the standardisation of each tool annotations or, in other words, of neutralising the effects of these discrepancies. The approach followed towards this aim is shown in Figure 18.

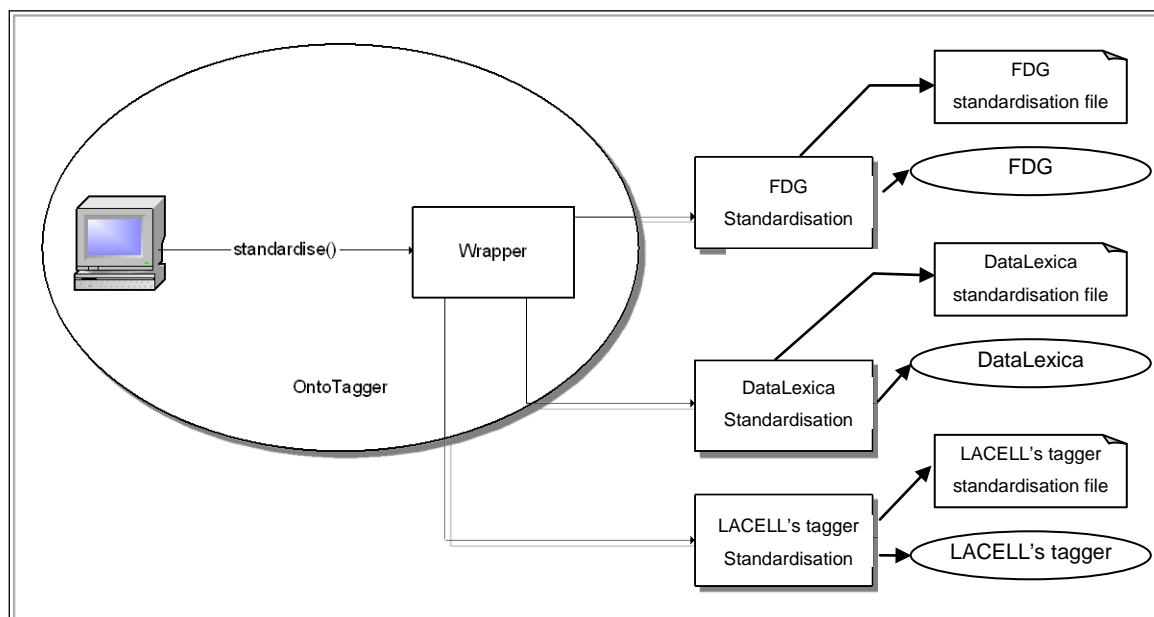


Figure 18: Approach followed for the standardisation of tools in OntoTagger

In this approach, before a linguistic annotation tool can be included in the configuration,

- 1) a setup file for the standardisation of its annotations must be created. This setup file simplifies the implementation of OntoTag’s configurations and enables their extensibility.
- 2) a new Java class for the standardisation of the new annotation tool must be programmed. This (new) Java class must implement a Java interface (defined within the configuration) that contains all the methods needed for (i) reading the input annotated document text; (ii) reading the setup file for standardisation; (iii) mapping the tool-dependent annotations onto standard annotations; and (iv) producing the output standard-annotated document. The new tool will communicate with the rest of OntoTag’s configuration by means of this (new) class. It must be

¹⁷⁸ A clitic Pronoun is a Pronoun that appears apposed (“stuck”) to a verbal form. EAGLES (1996a) recommendations refer to such a Pronoun as a Weak Pronoun. This is a frequent phenomenon in many Roman languages.

¹⁷⁹ See the details in Subsection 5.3.1, Syntactic Structure Combination.

compiled into a library and registered, as commented above, in order for the annotations of the tool to be conveniently standardised.

- **Manager Module for Annotation Combination Methods (MMACM):** A particular module was built to allow interchanging the combination methods which are used within OntoTagger. These combination methods are applied in the combination sub-phase of the merging phase, in order to merge the annotations for a given level coming from different linguistic annotation tools. The main functions implemented in the MMACM are (1) registering, updating and uninstalling a combination method from the system; and (2) enabling and disabling a combination method if necessary when annotating a document. All these combination methods can be executed when annotating a document with OntoTagger, and can operate at any level of linguistic description within the OntoTagger’s combination sub-phase. This makes the configuration more flexible, since (a) each combination method can implement a particular Artificial Intelligence reasoning model, suitable for the particular annotations being combined; and (b) the combination methods can be easily changed by simply changing this module. However, the development of this module did not advance much, since the main efforts were focused, for the sake of time and human resources, on the implementation of just one combination method. This combination method had to be suitable for the combination of each of the levels contemplated in its configuration according to their representation formalism. Consequently, the resulting combination method was implemented as a sort of *production system*¹⁸⁰.

In each one of these production systems,

1. the *working memory* contains (a) the whole set of annotation files (documents) being handled in the combination process in question; and (b) a small group of internal variables;
2. the *rule interpreter* is very simple, and was implemented *ad hoc*;
3. all its *rules* were assigned a number, and their numbering is the lowest rank criterion to decide which rule must be fired (*i.e.*, executed) first when more than one is triggered (*i.e.*, selected as a candidate to be executed). In this case, lower rule numbers have a higher priority. A higher rank criterion is firing first a rule that specifies or constitutes an exception to a more general one (called an *exception* for this reason). Exceptions are numerated too, but with a composite identifier, which determines their firing order as well. An exception identifier consists of: (i) the number of the rule they specify or with respect to which they constitute an exception; (ii) a particular number that differentiates it from the remaining exceptions to the same rule. In some levels, the execution of the rules can be

¹⁸⁰ A **production system** (or **production rule system**) is a type of computer system (or formalism), typically used in the field of Artificial Intelligence (AI). It consists primarily of a set of *rules* (or *productions*) of behaviour. Production systems also contain a sort of database, sometimes called its **working memory**, and a **rule interpreter**. Whereas the working memory stores the data that characterise the current state of the system, the rule interpreter must provide a mechanism for executing productions in order to achieve some goal for the system.

divided into two or more phases, where only one rule is fired (the one with the highest priority, according to these simple criteria). If this is the case, it will be conveniently indicated in the particular combination level.

- **Semantic Annotation Manager Module (SAMM):** In order to put an important load of semantics within OntoTagger's results, a particular module was developed for the production, integration and combination of semantic annotations, irrespective of their source or layer. On the one hand, first, this module performs named entity recognition, classification and tagging, according to the MUC and ACE tagsets. However, these tagsets were too coarse-grained for the domain of interest (entertainment, mainly film reviews, as mentioned above). Accordingly, they were complemented and further specified by means of a domain ontology (the Cinema Named Entities Ontology – CNEO) developed specifically for this prototype. Second, the SAMM sense-tags each content word found in the document being annotated by means of EuroWordNet synsets (*i.e.*, their identifier), and also with a reference to a suitable term in the CNEO when appropriate. And third, this module is responsible for the integration of the semantic information extracted from the linguistic tools included in the configuration¹⁸¹. On the other hand, the SAMM includes a process in charge of (a) the population of a domain ontology (the CNEO, in this case) with the instances identified within the process of named entity recognition and classification; and (b) the inclusion of new knowledge in the semantic lexicon implemented for this configuration, when appropriate. Hence, this last process can be regarded as a sort of linguistic knowledge-based machine learning process as well¹⁸².
- **Annotation Module (AM):** This is the most important module in OntoTagger, since it is responsible for the implementation of its configuration and for the overall, merged and hybrid (linguistic and ontological) annotation of documents. Obviously, the other modules aforementioned have a key role within the configuration, but the present module has to integrate them all and to annotate documents following the configuration shown in Figure 16. Thus, briefly, the aim of this module is to invoke, in the order determined by OntoTagger's configuration, the processes to (1) extract the plain text from the input document, leaving aside all its presentation mark-up (**Cleaning**); (2) input the plain text to each of the linguistic tools incorporated into the configuration and collect their outputs, which are documents annotated potentially at any Linguistic Level (**Tagging**); (3) map or translate the annotations of each tool into the terms of OntoTag's linguistic ontologies, so that they can be separated, compared, combined and integrated later on (**Standardisation**); (4) separate the annotations of each tool according to the

¹⁸¹ Such as the few semantic role annotations provided by Connexor's FDG Parser (source, goal, manner, etc. – see Table 92 in page 207) and the few semantic tags associated by LACELL's POS tagger to adverbs (manner, location, time, etc. – see Section 5.1.3 in page 221).

¹⁸² An example of the annotations resulting from this module is shown in Subsection 5.5.1 (in particular, see Table 139, Table 140 and Table 141).

level to which they belong (**Decanting**); (5) combine the annotations from the different tools at the same level, and determine the most accurate tag in each case for the combined annotation (**Combination**); and (6) put together the standardised and combined mono-level annotations into a standardised, combined and multi-level annotation (**Integration**). The outputs of this module are (1) an XML document, with the annotations associated to the input document. This document is not only machine-readable but also human-readable (for human annotators); and (2) a sort of translation of this XML document into (a) RDF(S) and (b) OWL.

- **Database Access Module (DBAM):** The prototype OntoTagger includes also a module that stores the annotations into the linguistic database ODELinger (Cantais-Sánchez, 2004). This alternative output of the system was included to facilitate a posterior statistical processing of these annotations¹⁸³. Therefore, this module includes the sequences of calls to the appropriate functions in ODELinger’s interface to store the annotation triples (unit-attribute-value) into the database.

After the main modules of OntoTagger have been introduced, those ones implementing the key points of the OntoTag model (that is, its most important phases) are described in the following subsections. First, the details concerning the combination module (MMACM) will be explained in Subsection 5.3. Then, in Subsection 5.4, a detailed description of the Semantic Annotation Manager Module (SAMM) has been included as well.

5.3. THE COMBINATION MODULE (MMACM) IN DETAIL

This section and its subsections present the different rules used for combining the results at the morphological, the morphosyntactic and the syntactic levels (or layers) of the linguistic annotation tools incorporated into OntoTagger. They are included in the production systems implemented within the Manager Module for Annotation Combination Methods (MMACM), introduced in the previous section (see page 216).

The *content of these combination rules (heuristics)* in OntoTagger (or the knowledge carried by them), in some cases it was determined empirically. This was done by checking and studying thoroughly the annotations generated by each of the linguistic annotation tools incorporated into the configuration. In some other cases, this content accounts for the differences among the purposes and the technologies underlying these linguistic tools. These differences usually make one of them outperform the others in certain occasions. In other cases, this content is the formalisation of some

¹⁸³ Database management systems include functions to analyse data in a most efficient way. Accordingly, it is more efficient to analyse statistically the annotations stored in the database than those include in the annotation files.

Spanish grammatical knowledge (such as agreement rules, for example, for the determination of the value of the `Gender` attribute of adjectives)¹⁸⁴.

The combination rules are presented here in the order in which they are processed, according to this architecture configuration (see Figure 16 –page 210– for details), namely (i) syntactic structure combination (Subsection 5.3.1), (ii) morphosyntactic category and lemma combination (Subsection 5.3.2), and (iii) morphological combination (Subsection 5.3.3)¹⁸⁵.

5.3.1. SYNTACTIC STRUCTURE COMBINATION

The set of rules included in the production system for syntactic combination consists mainly of word segmentation and token alignment rules, since only one tool performed real syntactic analysis (Connexor's FDG – dependency parsing). As mentioned above, there are several differences in the criteria applied for word segmentation in the different tools incorporated into the system. For example, one of the tools (LACELL's tagger) omits in its outputs those periods that come immediately after a parenthesis in the input text. Therefore, the execution of these combination rules must be prior to the combination of annotations at any other level, in order for the annotated documents to be processed synchronously and in parallel. As these rules are applied, all the resulting tokens are aligned. Eventually, a new output file will be generated for each tool, each containing the same number of tokens and in the same order. Besides, an additional new file is generated in this combination sub-phase. This additional file contains the same sequence of tokens, and the rest of the combined annotations which will be added to it later on (the morphosyntactic and the semantic ones).

The main hitches found at this stage are discussed next, together with the solutions implemented for these alignment errors. These solutions were empirically determined each time an alignment error appeared, after the annotations of the linguistic annotation tools had been carefully studied:

- **Alignment error #01:** FDG inserts many empty tokens, even though they may not appear in the input (clean) text. These empty tokens stand for formatting information, such an end of an empty line.
 - Solution implemented: locating and eliminating these empty tokens from the FDG output file.

¹⁸⁴ Several examples are shown all throughout Subsections 5.3.1, 5.3.2 and 5.3.3.

¹⁸⁵ The rules used for combining the results at the `Semantic Level` (and its different layers) can be better understood if they are explained together with the details of the Semantic Annotation Manager Module (SAMM). Therefore, they are not presented here, but in Section 5.4.2.1.

-
- **Alignment error #02:** LACELL's tagger does not annotate the character “ ’ ” (Simple Quotation Mark) when it appears immediately before a character “,” (Comma), “.” (Period), “;” (Semi-Colon) or “)” (Close Parenthesis).
 - Solution implemented: including an annotated “ ’ ” (Simple Quotation Mark) Token in the output file of LACELL's tagger when it appears in the annotated document of FDG and/or DataLexica, but not in the annotated document of LACELL's tagger.
 - **Alignment error #03:** LACELL's tagger does not annotate non-standalone apostrophes (“ ’ ”) (as in “Ocean's Eleven”, for example).
 - Solution implemented: The information about *Apostrophe* tokens outputted by the DataLexica standardiser, which does annotate apostrophes (“ ’ ”), is transferred to the annotated files of the rest of the tools.
 - **Alignment error #04:** LACELL's tagger includes a “.” (Period) where there is actually an end of paragraph instead of a “.” (Period) in the input (clean) file.
 - Solution implemented: removing the “.” (Period) tokens from the LACELL's tagger output file when the other tools do not contain a corresponding one in the same position.
 - **Alignment error #05:** Both FDG and LACELL's tagger group together, in just one Token, all those sequences of three or more “.” (*i.e.*, periods), interpreting them as an *ellipsis* (Suspension Points), but the annotated document coming from DataLexica contains a token for each “.” (Period).
 - Solution implemented: the annotations in the output files of FDG and of LACELL's tagger for sequences of “.” (*i.e.*, periods) are substituted with the annotations from DataLexica, since *ellipsis* (Suspension Points) do not appear as such within the EAGLES (1996a) recommendations.
 - **Alignment error #06:** FDG annotates as a single Token those multiword expressions attached by dashes (‘-‘), and makes no further processing of expressions including an *Apostrophe* in between (as in, for example, “Ocean's Eleven”). The rest of the tools annotate each element separately.
 - Solution implemented: the *Multiword Token* unit was created in OntoTag to cover these phenomena, that is, tokens consisting of more than one word. Each time this phenomenon is detected (a *Multiword Token* in FDG and a set of simple tokens in the other tools), a *Multiword Token* is included in all the input files. This *Multiword Token* consists of as many simple tokens as words and punctuation marks are detected by DataLexica or by LACELL's tagger.
-

- **Alignment error #07:** FDG often introduces a spurious Token in its output file when annotating a word that ends with what could be a clitic Pronoun but which, in fact, is not. For example, the Spanish word “*alas*” (≡ ‘wings’) could seem to be a word ended with the Spanish clitic Pronoun “*las*”, which is a clitic Pronoun standing for “them” (Feminine)¹⁸⁶. FDG annotates correctly the whole word, “*alas*”, and then introduces an extra (and spurious) Token for “*las*”, annotated as a clitic Pronoun.
 - Solution implemented: if a misalignment of this type is detected between FDG and DataLexica (*i.e.*, FDG annotates a clitic Pronoun after a word, but DataLexica does not), then the clitic Pronoun annotated by FDG is spurious and is removed from the FDG output file.
- **Alignment error #08:** LACELL’s tagger does not annotate the characters ‘¿’ (Open Question Mark) and ‘¡’ (Open Admiration Mark) when they head a new line in the input text.
 - Solution implemented: the annotation for these tokens is taken from the output file of any of the other tools.
- **Alignment error #09:** LACELL’s tagger annotation of commas (‘,’) is a bit erratic.
 - Solution implemented: whenever a misalignment of this type is detected (that is, LACELL’s tagger output lacks a Comma (‘,’) Token in a given position where the output files of the other tools include it), then a Comma Token is inserted in the LACELL’s output file.
- **Alignment error #10:** Both DataLexica and LACELL’s tagger annotate acronyms (such as “EE.UU.” in Spanish, which stands for U.S.A.) as several consecutive tokens separated by periods (“EE”, “.”, “UU”, “.” in this case), while FDG annotates them as a single Token, according to EAGLES (1996a) recommendations.
 - Solution implemented: transferring the annotation from FDG to the output files of the other tools.
- **Alignment error #11:** This type of misalignment is similar to the alignment error #06. Several multiword expressions separated by blank spaces, which are usually instances of named entities, are annotated by FDG as a single Token, while DataLexica and LACELL’s tagger treat them as several consecutive tokens. This is the case of the multiword expressions “Estados Unidos” (≈ United States) and “Reino Unido” (≈ United Kingdom).
 - Solution implemented: a Multiword Token is created, consisting of as many simple tokens as those annotated by DataLexica and LACELL’s tagger.

¹⁸⁶ Like in “*guárdalas*” (≡ “keep them”).

Additionally, in OntoTagger, once all the tokens in the output files of the linguistic annotation tool have been aligned, a process for the **recognition and annotation of Spanish prepositional locutions** (e.g. ‘a lo largo de’, which might be translated into ‘along’) is run over the different files. A list of prepositional locutions was extracted from Civit-Torruella (2003), and stored into a dedicated table of the semantic lexicon presented in Section 5.4.1. Whenever one of these locutions is detected in the input files, a `Multiword Token` is generated, and all of its individual tokens are included as constituent units (tokens) of this `Multiword Token`.

In conclusion, as already mentioned, in this combination sub-phase, a new document is created (the “Syn COMBINED” document in Figure 16), which shows the correspondences of each `Token` in the input text with its annotations coming from each tool. Each `Token` is assigned a particular URI in the XML “Syn COMBINED” document generated in this process. Therefore, the annotation for each `Token` in this new document must include (i) the `Word Form` of the `Token` (as it appears in the input file), accompanied by (ii) its URI in this combined document, and (iii) a group of references to the URIs of this `Token` in the output documents coming from the linguistic annotation tools. This last group of references is included in order to link together all the existing annotations for a `Token` of the input text (or document), each one coming from a different linguistic annotation tool. The resulting “Syn COMBINED” document will be enriched afterwards with the combined annotations of the rest of the levels incorporated into this configuration of OntoTag’s architecture.

5.3.2. MORPHOSYNTACTIC CATEGORY AND LEMMA COMBINATION

The aim of this combination sub-phase is to obtain a most precise morphosyntactic category and Lemma tagging of the input document. This is achieved by the comparison and combination of the annotations outputted by the linguistic tools integrated into the configuration.

The inputs of this combination sub-phase are the “L+C” decanted documents associated to these linguistic annotation tools, together with the “Syn COMBINED” document obtained in the Syntactic Combination Sub-Phase (see Figure 16). On the one hand, each “L+C” decanted document contains, for each `Token`, only its `Word Form`, its Lemma (L) and its morphosyntactic category (C). On the other hand, the “Syn COMBINED” document will be used as a guide in the combination of the morphosyntactic categories and lemmas, to retrieve the morphosyntactic category and Lemma tags assigned to each `Token` by each of the linguistic tools.

As for the most suitable order to combine the morphosyntactic category and the Lemma tags, it was established as follows. The possible Lemma tags for a `Token` fix its possible morphosyntactic category tags, and the possible morphosyntactic category tags of a `Token` fix its possible Lemma tags.

Additionally, the more accurate the morphosyntactic category tag of a `Token` is, the more accurate its Lemma tag will be, and *vice versa*. Therefore, both combinations are mutually dependent; however, they can be made one after the other in this configuration. Whereas some contextual knowledge can help disambiguate certain disagreements between tools when combining the morphosyntactic category tags of a `Token`, this is not so frequent a case in the combination of its Lemma tags. For this reason, it was decided to combine the morphosyntactic categories first and, then, combine the Lemma tags. Accordingly, the rules used for the combination of morphosyntactic categories are presented in the next subsection; the corresponding Lemma combination rules will be presented afterwards, in Subsection 5.3.2.2.

5.3.2.1 MORPHOSYNTACTIC CATEGORY COMBINATION

Before describing the rules used for the combination of morphosyntactic categories, it seems convenient to introduce the elements and the notation used to specify and represent them. In effect, the rules used in this combination sub-phase are a bit more complex than the ones applied to syntactic combination. Therefore, their specification requires the use of a mathematic and logic formalism to make it precise enough. Hence, in Subsection 5.3.2.1.1, the main mathematical concepts underlying this specification formalism will be surveyed; then, the elements in the notation of this particular type of combination rules will be described in Subsection 5.3.2.1.2; finally, the rules for morphosyntactic category combination will be enumerated and explained (when necessary) in Subsection 5.3.2.1.3.

5.3.2.1.1 Mathematical Terms Applied in the Notation

The main mathematical terms applied in the specification of the rules for morphosyntactic combination are reviewed in this subsection. The definitions of these terms have been taken mainly from Caballero-Roldán *et al.* (2007).

In mathematics, a **multiset** (or **bag**) is a generalization of a set, whose members (unlike those in sets) can have more than one membership. The total number of elements in a multiset, M , including repeated memberships, is the **cardinality** of the multiset (which can be notated as $|M|$), and the number of times an element belongs to the multiset is the **multiplicity** of that member. For example, in the multiset $M = \{x, x, x, y, y, z, z, z\}$, the multiplicities of the members x , y , and z are respectively 3, 2, and 3, and the cardinality of the multiset is 8.

Even though the order of enumeration of the members in a multiset is irrelevant, henceforth we might make a notation abuse, when necessary, to refer to each different member in the multiset by indexing it, as if in a programming array.

5.3.2.1.2 Description of the Notation

The specificities of the notation formalism for the rules applied to the combination of morphosyntactic categories within OntoTagger are described in this subsection. The terminology, (meta)variables and functions required for the representation of these rules are the following:

- **Terminology.**

- *Part of Speech (POS)*. This element of the notation refers to the major word category in a morphosyntactic tag. In other words, it refers to the only `Attribute` considered obligatory in the EAGLES (1996a) recommendations for morphosyntactic annotation. Its possible values are represented by: “N” (Noun), “V” (Verb), “AJ” (AdJective), “PD” (Pronoun/Determiner), “AT” (ArTicle), “AV” (AdVerb), “AP” (ApPosition), “C” (Conjunction), “NU” (NUmeral), “I” (Interjection), “U” (Unique/Unassigned), “R” (Residual) and “PU” (Punctuation Mark).
- *Underspecified category*. This term comes from EAGLES (1996a) as well. It stands for a category which is obtained amalgamating several alternative morphosyntactic categories (sharing at least their POS) into one, when (seemingly) all of them can be used to annotate a particular `Token`. All the alternative sub-specification attributes that have to be left underspecified are annotated between brackets (‘[’, ‘]’), and their respective alternative values are separated by the character ‘|’. For example, the underspecified tag NU[C|O] represents two alternative tags for a NUmeral (NU): NUC (Cardinal NUmeral) and NUO (Ordinal NUmeral).

- **(Meta)Variables.**

- $MSET_{CATEGORIES}$, $MSET_{CATEG1}$, $MSET_{CATEG2}$, $MSET_{OUTCATEGORIES}$, $MSET_{INCATEGORIES}$. Each of these variables designates a morphosyntactic category multiset (INCATEGORIES \equiv input of a procedure; OUTCATEGORIES \equiv output from a procedure).
- $MSET_{CATEGORIES}[index]$. This notation is used to select the morphosyntactic category referenced by *index* from the multiset of morphosyntactic categories $MSET_{CATEGORIES}$ (note the notation abuse aforementioned); the first possible value of *index* is 0 and, therefore, the last one is $|MSET_{CATEGORIES}| - 1$ (in Java, $sizeof(MSET_{CATEGORIES}) - 1$, which is the notation used here as well).
- $TEXT[offset]$. This metavariable refers to the `Word Form` associated to a `Token` that is separated by *offset* positions from the one whose morphosyntactic tag is being combined. Note that, if *offset* = 0, then the text in question is the one whose morphosyntactic tag is currently under combination; if *offset* = -1, then the text in question is the one associated to the morphosyntactic tag combined immediately before.

▪ **Functions.**

- *hasSymbol* (*category*, *symbol*) returns TRUE when the input *category* contains the input *symbol* (character array), starting at any position.
- *startsWith* (*category*, *symbol*) returns TRUE when the input *category* starts with the input *symbol* (character array).
- *getMax* ($MSET_{INCATEGORIES}$) returns the POS with the highest multiplicity in the multiset $MSET_{OUTCATEGORIES}$. If there exist several POSs fulfilling this property, all of them will be included in the output multiset, $MSET_{OUTCATEGORIES}$.
- *Category_Match*(*category*₁, *category*₂) returns TRUE when *category*₁ and *category*₂ share the same POS, and FALSE otherwise. For example:
 - *Category_Match*(“NP”, “NP”) = TRUE (NP = Noun, Proper)
 - *Category_Match*(“PDLP”, “PD”) = TRUE (PD = Pronoun/Determiner; PDLP = Pronoun/Determiner, reLative, Pronoun)
 - *Category_Match*(“V”, “VMP”) = TRUE (V = Verb; VMP = Verb, Main, Predicative)
 - *Category_Match*(“AVRN”, “AVD”) = TRUE (AV = AdVerb; AVRN = AdVerb, reLative, Non ‘Wh-’; AVD = AdVerb, Degree)
 - *Category_Match*(“AJ”, “NC”) = FALSE (AJ = AdJective; NC = Common Noun)
- *getCategories* (*category*) splits an underspecified category into the corresponding multiset of ambiguous alternative categories, $MSET_{OUTCATEGORIES}$. Returns $MSET_{OUTCATEGORIES}$. For example:
 - $NU[C|O] = \{NUC, NUO\}$
(NUC = NUmeral, Cardinal; NUO = NUmeral, Ordinal)
 - $PD[L|T|I]P = \{PDLP, PDTP, PDIP\}$
(PDLP = Pronoun/Determiner, reLative, Pronoun; PDTP = Pronoun/Determiner, inTerrogative, Pronoun; PDIP = Pronoun/Determiner, Indefinite, Pronoun)
- *Include*(*category*, $MSET_{INCATEGORIES}$) includes the *category* in the multiset $MSET_{INCATEGORIES}$ and returns the new resulting multiset, $MSET_{OUTCATEGORIES}$.

- *Assign*($MSET_{INCATEGORIES}$, $MSET_{OUTCATEGORIES}$) copies the multiset $MSET_{INCATEGORIES}$ to the multiset $MSET_{OUTCATEGORIES}$. If $MSET_{INCATEGORIES} = \emptyset$ (the empty set), then $MSET_{OUTCATEGORIES}$ is initialized in order to store a new value by means of a (series of) union operation(s).
- *Union*($MSET_{CATEG1}$, $MSET_{CATEG2}$) returns the multiset $MSET_{OUTCATEGORIES}$, consisting of the union of the two input multisets, $MSET_{CATEG1}$ and $MSET_{CATEG2}$.

5.3.2.1.3 Morphosyntactic Category Combination Rules

The rules for combining (standardised) morphosyntactic categories coming from different linguistic annotation tools are presented in this subsection. These morphosyntactic categories are assigned to the tokens in the input document by each linguistic tool incorporated into OntoTagger. As mentioned above, they are combined in order to select, for each token, the most accurate of them. This most accurate (combined) morphosyntactic category is the one assigned eventually to the Token by OntoTagger. This combination sub-phase consists of three different stages, namely (i) the selection of the candidate POSs (*i.e.*, discarding wrong categories and expanding underspecified tags); (ii) the selection of the POS with maximal multiplicity in the multiset of candidate POSs (*i.e.*, the one most supported by the tool annotations); and (iii) the distilment of the POS selected into a final, combined tag, as precise as possible with the information available. Each of these three stages has been encapsulated into a Java function, respectively, *Compare2Categories*, *CompareCategories* and *ResolveAndRefine* (described below). Each stage consists of a particular package of rules, where only one of them is to be triggered and fired at a time for a given token, after applying the selection criteria mentioned at the beginning of Section 5.3. After a rule is fired, the working memory of the package (*i.e.*, at least one of its internal variables) will be updated. This might cause (an)other rule(s) in the package to trigger and fire afterwards. The sub-phase ends when no rule remains triggered.

In what follows, it is assumed that (1) in each rule, the multiset $MSET_{CATEGORIES}$ is always initialised with the empty set (\emptyset) before any *Include* or *Union* operation is applied to it. (2) “RU” stands for Residual Unclassified, a category used in EAGLES (1996a) to describe tokens which can not be classified into any of the other morphosyntactic categories. In most cases, this is equivalent to say that the tool does not know how to tag the token. Therefore, this category matches any other nonempty category. (4) If a tool has not annotated a token with a morphosyntactic category tag, then it is attached the tag “null”. (5) ‘FAIL’ indicates that the disagreement between two categories is so notable that no matching is possible, and it must be solved in other stages of the combination.

- *Compare2Categories* ($category_1$, $category_2$): Returns a multiset, $MSET_{CATEGORIES}$, of candidate POSs, generated by firing one of the following rules (determined by the criteria aforementioned):
 - **RULE #0.1:** $category_1 = \text{“RU”} \rightarrow \text{Include}(category_2, MSET_{CATEGORIES})$

- **RULE #0.2:** $category_2 = "RU" \rightarrow Include(category_1, MSET_{CATEGORIES})$
- **RULE #1:** $category_1 = category_2 \rightarrow Include(category_1, MSET_{CATEGORIES})$
- **Exception #1.1:** $category_1 = category_2 \wedge hasSymbol(category_1, "[") \rightarrow Union(getCategories(category_1), MSET_{CATEGORIES})$
- **RULE #2:** $category_1 \neq category_2 \rightarrow "FAIL"$
- **Exception #2.1:** $category_1 \neq category_2 \wedge hasSymbol(category_1, "[") \wedge Category_Match(category_1, category_2) \rightarrow Union(getCategories(category_1), MSET_{CATEGORIES})$
- **Exception #2.2:** $category_1 \neq category_2 \wedge hasSymbol(category_1, "[") \wedge NOT(Category_Match(category_1, category_2)) \rightarrow \forall category \in getCategories(category_1), Union(Compare2Categories(category, category_2), MSET_{CATEGORIES})$
- **Exception #2.3:** $category_1 \neq category_2 \wedge hasSymbol(category_2, "[") \wedge Category_Match(category_2, category_1) \rightarrow Union(getCategories(category_2), MSET_{CATEGORIES})$
- **Exception #2.4:** $category_1 \neq category_2 \wedge hasSymbol(category_2, "[") \wedge NOT(Category_Match(category_2, category_1)) \rightarrow \forall category \in getCategories(category_2), Union(Compare2Categories(category, category_1), MSET_{CATEGORIES})$
- **Exception #2.5:** $category_1 \neq category_2 \wedge Category_Match(category_1, category_2) \rightarrow Include(category_1, MSET_{CATEGORIES})$
- **Exception #2.6:** $category_1 \neq category_2 \wedge Category_Match(category_2, category_1) \rightarrow Include(category_2, MSET_{CATEGORIES})$
- **CompareCategories** ($MSET_{INCATEGORIES}$): Returns the multiset of candidate POSs, $MSET_{OUTCATEGORIES}$, generated by firing the following rule, which selects the POS with the highest multiplicity (i.e., is the most likely to be correct) and all the tags associated to it:
 - **RULE #1:** $\forall a, b \in MSET_{INCATEGORIES} \rightarrow Assign(getMax(Compare2Categories(a, b)), MSET_{OUTCATEGORIES})$
- **ResolveAndRefine** ($MSET_{INCATEGORIES}$): after determining the (multi)set of tags which are most likely to be correct for a given token, it is the time for refining this (multi)set and try to return just one of the tags (categories) –the most accurate one– as a result ($MSET_{OUTCATEGORIES}$). The group of rules contained in this package was determined empirically, reviewing and comparing the results outputted by each tool (with the help of an *ad hoc* output file generated by OntoTagger). They solve many particular and sometimes inexplicable malfunctions of the linguistic tools; an example

of each of the problems solved, extracted from the corpus ODECorpus-Entertainment¹⁸⁷, is shown together with each rule in a dedicated table. Each of these tables includes (1) the token whose morphosyntactic category is combined using the associated rule, designated by the number of file (#FILE), (1.a) the token identifier (#TOKEN), built from its paragraph number, its sentence number within its paragraph and the number of the token within its sentence, separated by ‘_’; (2) its word form in the input file (TEXT); (3) the TOOL (DL = DataLexica; FDG = Connexor’s FDG; UM = LACELL’s POS tagger) which produced the CATEGORY TAG in the following column; (4) the combined tag (COMBINED TAG (STAGE 2)) entering the second stage; (5) the combined tag (COMBINED TAG (STAGE 3)) obtained after the application of the rule; (6) the CONTEXT of the token in the file where it appears; (7) a TRANSLATION of the Spanish phrase; and (8) when necessary, an appropriate COMMENT is also provided.

Thus, the set of rules applied in this stage and their illustrative examples are the following:

- **RULE #1:** $|MSET_{INCATEGORIES}| = 2 \wedge MSET_{INCATEGORIES}[1] = \text{“ATD”} \wedge MSET_{INCATEGORIES}[2] = \text{“ATI”} \wedge NOT \text{ (startsWith (TEXT[0], “u”))} \rightarrow Union(MSET_{INCATEGORIES}[1], MSET_{OUTCATEGORIES})$

Table 101: An example of application of RULE #1

CONTEXT		‘Guia del ocio España - la guia del ocio y entretenimiento’					
TRANSLATION		‘Spain leisure guide – the leisure and entertainment guide’					
COMMENTS		Both occurrences of “guia” contain an orthographic error (it should be written “guía” instead).					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	1_1_6	1_1_6_1	la	DL	PD[D I O P N]D (any kind of determiner)	ATD-ATI (definite article – indefinite article)	ATD
				DL	AT[D I] (any kind of article – either definite or indefinite)		
				DL	NC (common noun)		
				DL	PD (pronoun / determiner)		
			FDG	AT[D I] (Idem)			
			null (not annotated)	UM	null (no annotation given)		

¹⁸⁷ This Spanish corpus was created by the linguistic researchers of the OEG research group (<http://www.oeg-upm.net>). It was developed in the context of the project ContentWeb, within which the present Ph.D. dissertation was developed as well. It consists mainly of web pages on film and theatre play reviews, as well as restaurant commentaries. The foremost sources considered were la ‘Guía del ocio’ (<http://www.guiadelocio.es>) and ‘La Netro’ (<http://www.lanetro.com>).

- **EXCEPTION #1.1:** $| MSET_{INCATEGORIES} | = 2 \wedge MSET_{INCATEGORIES}[1] = \text{“ATD”} \wedge MSET_{INCATEGORIES}[2] = \text{“ATI”} \wedge \text{startsWith}(\text{TEXT}[0], \text{“u”}) \rightarrow \text{Union}(MSET_{INCATEGORIES}[2], MSET_{OUTCATEGORIES})$

Table 102: An example of application of EXCEPTION #1.1

CONTEXT		'El hijo de la novia, el derrumbe de un cuarentón'					
TRANSLATION		'The bride's son, the collapse of a man in his forties'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	84_1_4	84_1_4_1	un	DL	PD[D O P N]D	ATD-ATI	ATI
		35_1_76_1		DL	AT[D]		
		84_1_4_1		FDG	AT[D]		
		84_1_4_1	UM	null			

- **RULE #2:** $| MSET_{INCATEGORIES} | = 2 \wedge MSET_{INCATEGORIES}[1] = \text{“NP”} \wedge MSET_{INCATEGORIES}[2] = \text{“NC”} \rightarrow \text{Union}(MSET_{INCATEGORIES}[1], MSET_{OUTCATEGORIES})$

Table 103: An example of application of RULE #2

CONTEXT		'VIDOCQ Fantasía y muerte en el París de 1830'					
TRANSLATION		'VIDOCQ Fantasy and death in the Paris of 1830'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	5_1_6	5_1_6_1	París	DL	RU	NP-NC (proper noun – common noun)	NP
		8_1_6_1		FDG	NP		
		1_1_6_1		UM	NC		

- **RULE #3:** $| MSET_{INCATEGORIES} | = 2 \wedge MSET_{INCATEGORIES}[1] = \text{“NC”} \wedge MSET_{INCATEGORIES}[2] = \text{“NP”} \rightarrow \text{Union}(MSET_{INCATEGORIES}[1], MSET_{OUTCATEGORIES})$

Table 104: An example of application of RULE #3

CONTEXT		'Página Web'					
TRANSLATION		'Web page'					
COMMENTS		This is an isolated piece of text in the input file – an HTML link to the homepage of the film.					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	15_1_2	15_1_2_1	Web	DL	RU	NC-NP	NC
		9_3_9_1		FDG	NC		
		1_11_2_1		UM	NP		

- **RULE #4:** $| MSET_{INCATEGORIES} | = 2 \wedge MSET_{INCATEGORIES}[1] = \text{“AVGN”} \wedge MSET_{INCATEGORIES}[2] = \text{“AVRN”} \rightarrow \text{Union}(MSET_{INCATEGORIES}[1], MSET_{OUTCATEGORIES})$

Table 105: An example of application of RULE #4

CONTEXT		'trata de reconstruir hacia atrás la vida de tan enigmático personaje'					
TRANSLATION		'tries to reconstruct backwards the life of such an enigmatic character'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	18_1_9	18_1_9_1	atrás	DL	AV	AVGN-AVRN (general non-Wh- adverb – pronominal non-Wh- adverb)	AVGN
		10_3_9_1		FDG	AV		
		1_14_9_1		UM	AV[G R]N		

- **RULE #5:** $| MSET_{INCATEGORIES} | = 2 \wedge MSET_{INCATEGORIES}[1] = "PDLP" \wedge startsWith(MSET_{INCATEGORIES}[2], "CS") \rightarrow Union(MSET_{INCATEGORIES}[1], MSET_{OUTCATEGORIES})$

Table 106: An example of application of RULE #5

CONTEXT		'un thriller gótico realizado en vídeo digital que une intriga, fantasía y realidad'					
TRANSLATION		'a gothic thriller filmed in digital video which brings together suspense, fantasy and reality'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	6_1_12	6_1_12_1	que	DL	PD	PDLP-CSS (relative pronoun – substantive subordinating conjunction)	PDLP
				DL	C[C S] (conjunction, either coordinating or subordinating)		
				FDG	PDL[P D]		
				UM	CSS		

- **RULE #6:** $| MSET_{INCATEGORIES} | = 2 \wedge MSET_{INCATEGORIES}[1] = "NC" \wedge MSET_{INCATEGORIES}[2] = "CC" \rightarrow Union(MSET_{INCATEGORIES}[2], MSET_{OUTCATEGORIES})$

Table 107: An example of application of RULE #6

CONTEXT		'estudiando después lengua y literatura francesa e iniciando una carrera como actriz'					
TRANSLATION		'studying French Language and Literature later on, and starting her career as an actress'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	28_1_20	28_1_20_1	e	DL	NC	NC-CC	CC
				DL	C[C S]		
				FDG	CC		
				UM	NC		

- **RULE #7:** $| MSET_{INCATEGORIES} | = 2 \wedge MSET_{INCATEGORIES}[1] = "NC" \wedge MSET_{INCATEGORIES}[2] = "AV" \rightarrow Union(MSET_{INCATEGORIES}[2], MSET_{OUTCATEGORIES})$

Table 108: An example of application of RULE #7

CONTEXT		'trata de reconstruir hacia atrás la vida de tan enigmático personaje'					
TRANSLATION		'tries to reconstruct backwards the life of such an enigmatic character'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	18_1_13	18_1_13_1	tan	DL	NC	NC-AV (common noun - adverb)	AV
				DL	AV		
				FDG	AV		
				UM	NC		

- **RULE #8:** $| MSET_{INCATEGORIES} | = 2 \wedge MSET_{INCATEGORIES}[1] = "VM" \wedge MSET_{INCATEGORIES}[2] = "APR" \rightarrow Union(MSET_{INCATEGORIES}[2], MSET_{OUTCATEGORIES})$

Table 109: An example of application of RULE #8

CONTEXT		'el difuminado límite entre lo imaginario y lo real'					
TRANSLATION		'the fuzzy limit between what is imaginary and what is real'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	18_1_26	18_1_26_1	entre	DL	V	VM-APR (main verb - preposition)	APR
				DL	V		
				DL	APR		
				UM	APR		

- **RULE #9:** $| MSET_{INCATEGORIES} | = 2 \wedge MSET_{INCATEGORIES}[1] = "NC" \wedge MSET_{INCATEGORIES}[2] = "PD" \rightarrow Union(MSET_{INCATEGORIES}[2], MSET_{OUTCATEGORIES})$

Table 110: An example of application of RULE #9

CONTEXT		'más tarde, cuando cada uno de ellos está a punto de casarse'					
TRANSLATION		'afterwards, when each one of them is close to getting married'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
3	16_1_11	16_1_11_1	uno	DL	V	NC-PD	PD
				DL	V		
				DL	NC		
		DL		PD			
		FDG		PD			
		1_5_11_1	UM	NC			

- **RULE #10:** $| MSET_{INCATEGORIES} | = 3 \wedge MSET_{INCATEGORIES}[1] = "ATD" \rightarrow Union(MSET_{INCATEGORIES}[1], MSET_{OUTCATEGORIES})$

Table 111: An example of application of RULE #10

CONTEXT		'Título de la película: Vidocq'					
TRANSLATION		'Title of the film: Vidocq'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	7_1_3	7_1_3_1	la	DL	PD[D O P N]D	ATD-ATI-NC	ATD
				DL	AT[D]		
				DL	NC		
		DL		PD			
		FDG		AT[D]			
		9_1_27_1	UM	NC			

- **RULE #11:** $| MSET_{INCATEGORIES} | = 3 \wedge MSET_{INCATEGORIES}[1] = "VM" \wedge MSET_{INCATEGORIES}[2] = "NC" \wedge MSET_{INCATEGORIES}[3] = "AJ" \rightarrow Union(MSET_{INCATEGORIES}[2], MSET_{OUTCATEGORIES})$

Table 112: An example of application of RULE #11

CONTEXT		'un terrible asesino apodado "El Alquimista" '					
TRANSLATION		'a terrible murderer whose nickname was "The Alchemist" '					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	17_1_52	17_1_52_1	asesino	DL	V	VM-NC-AJ	NC
				DL	NC		
				DL	AJ (adjective)		
		FDG		AJ			
		FDG		VM			
		1_13_52_1	UM	NC			

- **RULE #12:** $| MSET_{INCATEGORIES} | = 3 \wedge MSET_{INCATEGORIES}[1] = "NC" \wedge MSET_{INCATEGORIES}[2] = "V" \wedge MSET_{INCATEGORIES}[3] = "AJ" \rightarrow Union(MSET_{INCATEGORIES}[2], MSET_{OUTCATEGORIES})$

Table 113: An example of application of RULE #12

CONTEXT		'Sin noticias de Dios, ángel y demonio con las caras cambiadas '					
TRANSLATION		'No news from God, angel and devil with swapped faces'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	82_1_7	82_1_7_1	cambiadas	DL	NC	NC-V-AJ	V
				DL	V		
				DL	V		
		35_1_66_1	FDG	AJ			
		82_1_7_1	null	UM	null		

- **RULE #13:** $| MSET_{INCATEGORIES} | = 3 \wedge MSET_{INCATEGORIES}[1] = "PU09" \wedge MSET_{INCATEGORIES}[2] = "PU10" \wedge MSET_{INCATEGORIES}[3] = "PU13" \wedge TEXT[-1] = "(" \rightarrow Union(MSET_{INCATEGORIES}[1], MSET_{OUTCATEGORIES})$

Table 114: An example of application of RULE #13

CONTEXT		'Michael Winterbottom ('With or without you' y 'Wonderland')					
TRANSLATION		'Michael Winterbottom ('With or without you' and 'Wonderland')					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
2	19_1_4	19_1_4_1	,	DL	PU[09 10 13]	PU09-PU10-PU13 (simple quotation mark (open) – simple quotation mark (close) – apostrophe)	PU09
		12_1_4_1		FDG	PU[09 10 13]		
		19_1_4_1	null	UM	null		

- **RULE #14:** $| MSET_{INCATEGORIES} | = 3 \wedge MSET_{INCATEGORIES}[1] = "PU09" \wedge MSET_{INCATEGORIES}[2] = "PU10" \wedge MSET_{INCATEGORIES}[3] = "PU13" \rightarrow Union(MSET_{INCATEGORIES}[2], MSET_{OUTCATEGORIES})$

Table 115: An example of application of RULE #14

CONTEXT		'la producción francesa 'Vidocq', una apasionante intriga policiaca'					
TRANSLATION		'the French production 'Vidocq', an exciting crime suspense (one)'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	16_1_15	16_1_15_1	,	DL	PU[09 10 13]	PU09-PU10-PU13	PU10
		10_1_15_1		FDG	PU[09 10 13]		
		16_1_15_1	null	UM	null		

- **RULE #15:** $| MSET_{INCATEGORIES} | > 5 \wedge startsWith (MSET_{INCATEGORIES}[sizeof(MSET_{INCATEGORIES})-1], "NU") \rightarrow Union(MSET_{INCATEGORIES}[|MSET_{INCATEGORIES}|-1], MSET_{OUTCATEGORIES})$

Table 116: An example of application of RULE #15

CONTEXT		'Nacida en Valladolid en 1973, sus primeros trabajos fueron como modelo'					
TRANSLATION		'Born in Valladolid in 1973, she worked first as model'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	28_1_8	28_1_8_1	primeros	DL	PD[D I O P N]D	PDDD-PDID-PDOD-PDPD-PDND-ATD-ATI-NUO-AJ (demonstrative determiner – indefinite determiner – possessive determiner – partitive determiner – numeral determiner – definite article – indefinite article – ordinal numeral – adjective)	NUO
		19_1_8_1		DL	AT[D I]		
		1_23_10_1		FDG	NUO		
				UM	AJ		

- **RULE #16:** $| MSET_{INCATEGORIES} | > 5 \wedge MSET_{INCATEGORIES}[1] = "PDDD" \wedge MSET_{INCATEGORIES}[MSET_{INCATEGORIES}|-1] = "V" \rightarrow Include("ATI", MSET_{OUTCATEGORIES})$

Table 117: An example of application of RULE #16

CONTEXT		'A mi madre le gustan las mujeres, Una familia poco tradicional'					
TRANSLATION		'My mother likes women, hardly a traditional family'					
COMMENTS		The capital letter in 'Una', after a comma, can be considered another orthographic error.					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
1	62_1_1	62_1_1_1	Una	DL	PD[D I O P N]D	PDDD-PDID-PDOD-PDPD-PDND-ATD-ATI-V-NP	ATI
				DL	AT[D I]		
				DL	PD		
				DL	PD[D I O P N]D		
				DL	AT[D I]		
				DL	V		
				DL	V		
				DL	PD		
		FDG	NP				
		33_1_23_1					
		62_1_1_1	null	UM	null		

- **RULE #17:** $| MSET_{INCATEGORIES} | > 5 \wedge MSET_{INCATEGORIES}[1] = "AJ" \wedge MSET_{INCATEGORIES}[MSET_{CATEGORIES}|-1] = "PDPD" \rightarrow Union(MSET_{INCATEGORIES}[1], MSET_{OUTCATEGORIES})$

Table 118: An example of application of RULE #17

CONTEXT		'Aunque sin dudar de la pericia de director y actores, o del encanto mismo de la historia'					
TRANSLATION		'Though doubting not of the director's and the actors' skills, or even of the very charm of the story'					
#FILE	#TOKEN	#WORD	TEXT	TOOL	CATEGORY TAG	COMBINED TAG (STAGE 2)	COMBINED TAG (STAGE 3)
3	18_1_15	18_1_15_1	mismo	DL	AJ	AJ-PDDD-PDID-PDOD-PDPD-PDND	AJ
				DL	PD		
				DL	PD[D I O P N]D		
				DL	AT[D I]		
		FDG	PD				
		1_9_15_1		UM	AJ		

The rules applied within the morphosyntactic category combination sub-phase of OntoTagger have been presented in this subsection. As explained, they are used in the comparison and combination of the different morphosyntactic categories assigned to each particular token by the different linguistic tools integrated into the system. After this sub-phase has been executed, each token has been assigned a most correct morphosyntactic tag. The next step towards completing the combination of the annotations at the `MorphoSyntactic Level` is combining the `Lemma` combination, which is explained in the following subsection.

5.3.2.2 LEMMA COMBINATION

In OntoTagger's configuration, after the morphosyntactic categories assigned by the different linguistic tools to each token in the input text have been combined, it is the time for combining the lemmas that these tools assigned to it as well. However, as with morphosyntactic category combination rules, before describing the rules used for `Lemma` combination, it seems convenient to introduce the elements and the notation used to specify and represent them. Thus, in Subsection 5.3.2.2.1, it will be surveyed the main mathematical concepts underlying the notation used; then, the elements in the notation of this particular type of combination rules will be described in Subsection 5.3.2.2.2; finally, the rules for `Lemma` combination will be enumerated and explained (when necessary) in Subsection 5.3.2.2.3.

5.3.2.2.1 Mathematical Terms Applied in the Notation

The main mathematical terms applied in the specification of the rules for `Lemma` combination are the same presented in Subsection 5.3.2.1.1. They are not included here for the sake of brevity.

5.3.2.2.2 Description of the Notation

The notation used for the specification of the `Lemma` combination rules is very similar to the one used for morphosyntactic category combination rules, excepting that it has to be extended with some more variables, functions and procedures.

The first important issue in this description is a kind of formal generalisation of the results obtained in the development of OntoTagger. Some of the rules in the combination of lemmas were determined taking into account not a particular tool incorporated into the system, but the technology applied in its construction. Therefore, the notation concerning the three different tools incorporated into this OntoTag's architecture configuration is:

- **NDLPT**: Non-Disambiguating Lemma and POS Tagger (a morphological analyser: **DataLexica**).

- **PLPMT**: Pure Lemma, POS and Morphological Tagger (**LACELL’s POS tagger**)
- **LPMT**: Lemma, POS and Morphological Tagger, and Parser (**FDG**)
- **TOOLS**: The whole set of tools integrated within an architecture configuration. In this case, $TOOLS = \{NDLPT, PLPMT, LPMT\}$
- **MSET_{TOOLS}**: Designates the multiset of linguistic tools incorporated into the configuration that generated a Lemma tag for a Token. The possible members of $MSET_{TOOLS}$ are the members of the set $TOOLS$, and their multiplicity is any natural number. If a tool has not assigned a Lemma tag to a token, then it is assigned the Lemma tag “null”.

Once these primary elements in the notation have been introduced, some others can be defined from them as well:

- **(Meta)Variables.**
 - CAT_{NDLPT} , CAT_{PLPMT} , CAT_{LPMT} . These variables refer to the (multi)set of morphosyntactic category tag(s) assigned by the corresponding linguistic tool to a certain Token. As shown in the previous subsection, each of these (multi)sets can contain more than one tag and each tag can have more than one membership in the multiset.
 - CAT_t . This metavariable designates any of the three variables described previously, depending on the value that takes t eventually, where $t \in TOOLS$.
 - CAT_{COMB} . This variable designates the combined morphosyntactic category of a Token.
 - $lemma_{COMB}$. This variable refers to the (final) combined Lemma tag of a Token, that is, it is a particular result of this combination sub-phase.
 - $lemma_t$. This metavariable designates the (set of) Lemma tag(s) assigned to a Token by the tool t , where $t \in TOOLS$.
- **Functions.**
 - $Multiplicity(tool, MS_{TOOLS})$ returns the multiplicity of the *tool* within the multiset MS_{TOOLS} .
 - $isApocope(lemma, category)$ returns TRUE when the input parameter *lemma* can be considered the apocopation of another Lemma, according to the input *category* (supposedly the one associated to the *lemma*); returns FALSE otherwise. For example¹⁸⁸:
 - $isApocope(“gran”, AJ) = TRUE$
 - $isApocope(“grande”, AJ) = FALSE$

¹⁸⁸ “Gran” is an apocopation of “grande”; “grande” stands for large, big, great, tall (amongst other secondary senses).

- *Category_Match*(*category*₁, *category*₂) returns TRUE when *category*₁ and *category*₂ share the same POS, and FALSE otherwise (some examples have been included in above, in Section 5.3.2.1.2).
- *Category_Equal*(*category*₁, *category*₂) returns TRUE when the values of *category*₁ and *category*₂ are exactly the same, and FALSE otherwise.
- *hasValue*(*lemma*, *category*, *attribute*, *value*) returns TRUE when the input parameter *lemma* is correct according to the input *value* tag for the input *attribute* of the input *category*, supposedly all of them associated to the same token by a given tool; returns FALSE otherwise. For example¹⁸⁹:
 - *hasValue*(“mayor”, AJ, Degree, COMPARATIVE) = TRUE
 - *hasValue*(“paupérrimo”, AJ, Degree, SUPERLATIVE) = TRUE
 - *hasValue*(“mejor”, AJ, Degree, POSITIVE) = FALSE
- **Procedures.**
 - *Assign*(*lemma*₁, *lemma*₂) stores the Lemma tag(s) in the input parameter, *lemma*₂, into the output parameter, *lemma*₁.

5.3.2.2.3 Lemma Combination Rules

The rules for Lemma combination are presented and described in this subsection. They were determined empirically, checking and comparing the annotations coming from each linguistic tool incorporated into the configuration. This was done with the help of an output file generated *ad hoc* by OntoTagger, which summarises the discordances between these Lemma tags. These rules for Lemma combination solve many particular (and sometimes inexplicable) malfunctions of the linguistic annotation tools.

To begin with, a sort of general scheme (or meta-rule) for Lemma combination was found when studying the output file containing the Lemma tagging discordances aforementioned. It can be specified as follows:

- *LemmaCombination*(OUT *lemma*₁, IN *lemma*₂):
 - IF there is **only one value** stored in the input parameter *lemma*₂ THEN
 - *lemma*₁ is assigned the unique value in *lemma*₂

¹⁸⁹ “Mayor” stands for ‘older’ or ‘bigger’; ‘paupérrimo’ stands for ‘poorest’; “major” stands for ‘better’.

- ELSE (* when there is more than one value stored in $lemma_2$ *)
 - IF the token in question has got a **verbal** (sub)category tag THEN:
 - IF $Category_Match(CAT_{COMB}, CAT_{LPMT})$ THEN
 - $lemma_1$ is assigned the value in $lemma_{LPMT}$
 - ELSIF $Category_Match(CAT_{COMB}, CAT_{PLPMT})$ THEN
 - $lemma_1$ is assigned the value in $lemma_{PLPMT}$
 - ELSE
 - $lemma_1$ is assigned the first irreflexive Lemma in $lemma_{NDLPT}$
 - ELSE
 - $lemma_1$ is assigned the first value stored in $lemma_{NDLPT}$.

In the formulation of the previous general scheme for Lemma combination it was taken into account that

- 1) Most frequently, once the morphosyntactic category has been correctly determined, the best choice as for its Lemma is the one associated to that morphosyntactic category in the annotations of the *NDLPT* (except when it is a verbal (sub)category). In effect, the annotations coming from the *NDLPT* are the most accurate ones once a *Word Form* and its corresponding morphosyntactic category have been fixed. For this reason, by default, $lemma_2$ is assigned the lemmas coming from the *NDLPT*.
- 2) The Lemma tagging of the *NDLPT* for verbs is highly ambiguous, since it often includes at least two very similar lemmas, namely the one associated to a reflexive use of the verb and the one associated to its non-reflexive use. Accordingly, the Lemma tag chosen for a token in this case is the one assigned to it by a tool (either the *LPMT* or the *PLPMT*) that also assigned to it a correct morphosyntactic category (which should be already included in CAT_{COMB}).

The different rules for Lemma combination were developed according to the general scheme described above. Each of the resulting Lemma combination rules is presented next, in a dedicated table, together with an example of application, extracted from the corpus ODECORPUS-Entertainment. In each of these tables, it has been included (1) the *Token* whose Lemma is combined using the associated rule, designated by (1.a) the number of file (*#FILE*), (1.b) the *Token* identifier (*#TOKEN*), consisting of its paragraph number, its sentence number within its paragraph and its number of token within its sentence, separated by ‘_’; (2) its *Word Form* in the input document (*TEXT*); (3) the *TOOL* (DL = DataLexica; FDG = Connexor’s FDG; UM = LACELL’s POS tagger) which produced

the TOOL CATEGORY in the following column; (4) its combined morphosyntactic category tag obtained previously (COMBINED CATEGORY); (5) the COMBINED LEMMA obtained after the application of the rule; (6) the CONTEXT of the Token in the input file where it appears; (7) a TRANSLATION into English of this Spanish context Phrase; and (8) some appropriate COMMENTS, when necessary. Thus, the set of rules applied in this stage and their illustrative examples are the following:

- **RULE #0:** $\exists t \in MSET_{TOOLS} \mid (Multiplicity(t, MSET_{TOOLS}) = n \wedge |MSET_{TOOLS}| = n) \wedge NOT (Category_Equal("null", CAT_t)) \rightarrow Assign(lemma_{COMB}, lemma_t)$

Table 119: An example of application of RULE #0

CONTEXT		"Crítica de F. Méndez-Leite"						
TRANSLATION		"F. Méndez-Leite's commentary"						
COMMENTS		A POS tagging error that could not be solved with the linguistic tools involved.						
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA
1	42_2_1	42_2_1_1	Méndez	DL	RU	méndez	RU	Méndez
				FDG	null	null		
				UM	null	null		

- **RULE #1:** $\exists cat \in CAT_{NDLPT} \mid Category_Match(cat, CAT_{COMB}) \rightarrow Assign(lemma_{COMB}, lemma_{cat})$

Table 120: An example of application of RULE #1

CONTEXT		"Crítica de F. Méndez-Leite"						
TRANSLATION		"F. Méndez-Leite's commentary"						
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA
1	42_1_5	42_1_5_1	.	DL	PU01	.	PU01	.
		30_1_5_1	.	FDG	PU01	.		
		42_1_5_1	null	UM	null	null		

- **EXCEPTION #1.0:** $Category_Equal("RU", CAT_{COMB}) \rightarrow Assign(lemma_{COMB}, TEXT)$

Table 121: An example of application of EXCEPTION #1.0

CONTEXT		"El hijo de la novia, el derrumbe de un cuarentón "						
TRANSLATION		"The bride's son, the collapse of a man in his forties "						
COMMENTS		POS combination failure.						
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA
1	84_1_5	84_1_5_1	cuarentón	DL	AJ	cuarentón	RU	cuarentón
		35_1_77_1	cuarentón	FDG	NC	cuarentón		
		84_1_5_1	null	UM	null	null		

- **EXCEPTION #1.1:** $\exists cat1 \in CAT_{NDLPT} \mid Category_Match(cat1, CAT_{COMB}) \wedge Category_Match(CAT_{COMB}, "AJ") \wedge isApocope(lemma_{cat1}, CAT_{COMB}) \wedge \exists cat2 \in CAT_{LPMTPT} \mid Category_Match(cat2, CAT_{COMB}) \rightarrow Assign(lemma_{COMB}, lemma_{cat2})$

Table 122: An example of application of EXCEPTION #1.1

CONTEXT		"La gran triunfadora del pasado Festival de Sitges"						
TRANSLATION		"The great triumphant (film) in the last Sitges Festival"						
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA
1	16_1_2	16_1_2_1	gran	DL	AVGN	gran	AJ	grande
				DL	AJ	gran		
		10_1_2_1		FDG	AJ	grande		
		1_12_2_1		UM	AJ	grande		

- EXCEPTION #1.2:** $\exists cat1 \in CAT_{NDLPT} \text{ Category_Match}(cat1, CAT_{COMB}) \wedge \text{Category_Match}(CAT_{COMB}, "AJ") \wedge NOT(\text{hasValue}(\text{lemma}_{cat1}, CAT_{COMB}, DEGREE, POSITIVE)) \wedge \exists cat2 \in CAT_{LPMTP} | \text{Category_Match}(cat2, CAT_{COMB}) \rightarrow \text{Assign}(\text{lemma}_{COMB}, \text{lemma}_{cat2})$

Table 123: An example of application of EXCEPTION #1.2

CONTEXT		"El Señor de los Anillos, la mayor aventura jamás filmada"						
TRANSLATION		"The Lord of the Rings, the greatest adventure ever filmed"						
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA
1	70_1_2	70_1_2_1	mayor	DL	NC	mayor	AJ	grande
				DL	AVGN	mayor		
				DL	AJ	mayor		
				FDG	AJ	grande		
		70_1_2_1	null	UM	null	null		

- SPECIAL CASE #1.1:** $| CAT_{NDLPT} | > 1 \wedge \exists cat1, cat2 \in CAT_{NDLPT} | [\text{Category_Match}(cat1, CAT_{COMB}) \wedge \text{hasValue}(\text{lemma}_{cat1}, cat1, GENDER, MASCULINE) \wedge \text{Category_Match}(cat2, CAT_{COMB}) \wedge NOT(\text{hasValue}(\text{lemma}_{cat2}, cat2, GENDER, MASCULINE))] \rightarrow \text{Assign}(\text{lemma}_{COMB}, \text{lemma}_{cat1})$

Table 124: An example of application of SPECIAL CASE #1.1

CONTEXT		"Vidocq', una apasionante intriga policiaca basada en un personaje real"							
TRANSLATION		"Vidocq', an exciting crime suspense based upon a real character"							
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA	
1	16_1_17	16_1_17_1	una	DL	PD[D] O P N]D	un	ATI	un	
				DL	AT[D]I]	un			
				DL	PD	una			
				DL	PD[D] O P N]D	una			
				DL	AT[D]I]	una			
				DL	V	unir			
				DL	V	unirse			
				DL	PD	uno			
				10_1_17_1	FDG	AT[D]I]			uno
				1_12_16_1	UM	ATI			uno

- **RULE #2:** $\forall cat1 \in CAT_{NDLPT} \text{ NOT } (Category_Match(cat1, CAT_{COMB})) \wedge \exists cat2 \in CAT_{LPMT} | Category_Match(cat2, CAT_{COMB}) \rightarrow Assign(lemma_{COMB}, lemma_{c2})$

Table 125: An example of application of RULE #2

CONTEXT		"Crítica de F. Méndez-Leite"						
TRANSLATION		"F. Méndez-Leite's commentary"						
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA
1	42_1_2	42_1_2_1	Crítica	DL	RU	crítica	NC	crítica
		30_1_2_1		FDG	NC	crítica		
		42_1_2_1	null	UM	null	null		

- **EXCEPTION #2.1:** $|CAT_{NDLPT}| = 1 \wedge Category_Equal("NP", CAT_{COMB}) \wedge \forall cat1 \in CAT_{NDLPT} Category_Equal(cat1, "RU") \rightarrow Assign(lemma_{COMB}, TEXT)$

Table 126: An example of application of EXCEPTION #2.1

CONTEXT		"Amélie, Mejorando vidas ajenas"						
TRANSLATION		"Amélie, improving other people's lives"						
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA
1	85_1_1	85_1_1_1	Amélie	DL	RU	amélie	NP	Amélie
		35_1_78_1		FDG	NP	amélie		
		85_1_1_1	null	UM	null	null		

- **EXCEPTION #2.2:** $|CAT_{NDLPT}| = 1 \wedge Category_Equal("NC", CAT_{COMB}) \wedge \forall cat1 \in CAT_{NDLPT} Category_Equal(cat1, "RU") \rightarrow Assign(lemma_{COMB}, TEXT)$

Table 127: An example of application of EXCEPTION #2.2

CONTEXT		"Página Web"						
TRANSLATION		"Web page"						
COMMENTS		This is an isolated piece of text in the input file, a link to the film homepage.						
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA
1	15_1_2	15_1_2_1	Web	DL	RU	web	NC	web
		9_3_9_1		FDG	NC	web		
		1_11_2_1		UM	NP	Web		

- **RULE #3:** $\forall cat1 \in CAT_{NDLPT} \text{ NOT } (Category_Match(cat1, CAT_{COMB})) \wedge \forall cat2 \in CAT_{LPMT} \text{ NOT } (Category_Match(cat2, CAT_{COMB})) \wedge \exists cat3 \in CAT_{PLPMT} | Category_Match(cat3, CAT_{COMB}) \rightarrow Assign(lemma_{COMB}, lemma_{c3})$

Table 128: An example of application of RULE #3

CONTEXT		"un personaje real que vivió en el convulso París de 1830."						
TRANSLATION		"a real character who lived in the convulsed Paris of 1830."						
COMMENTS		A rather redundant rule - in most cases, other rule with higher priority is triggered as well, and fired instead.						
#FILE	#TOKEN	#WORD	TEXT	TOOL	TOOL CATEGORY	TOOL LEMMA	COMBINED CATEGORY	COMBINED LEMMA
1	16_1_26	16_1_26_1	que	DL	PD	que	PDL-P-CSS	que
				DL	C[C S]			
				FDG	PDL[P D]	que		
				UM	CSS	que		

The rules applied within the Lemma combination sub-phase of OntoTagger have been presented in this subsection. As explained, they are used in the comparison and combination of the different lemmas assigned to each particular token by the different linguistic tools integrated into the system. After this sub-phase has been executed, each token has been assigned a most correct morphosyntactic and a Lemma tags. For the morphosyntactic annotation combination to be completed, it is necessary to combine the different morphological attribute values assigned by the different tools to the tokens in the input document. How this was done in OntoTagger is explained in the following subsection.

5.3.3. MORPHOLOGICAL COMBINATION

In this OntoTag's architecture configuration, the morphological combination sub-phase takes place after the Lemma combination has been completed. Thus, the document containing the re-numbered and synchronised sequence of paragraphs, sentences and tokens (the "Syn COMBINED" document in Figure 16, included in page 210), has already been enriched with the standardised and combined morphosyntactic and Lemma annotations. The resulting document, is the "L+POS COMBINED" document in the aforementioned Figure 16. This document is supplied to this sub-phase as an input, together with the decanted files containing the morphological annotations of the input document coming from the different linguistic tools integrated into the configuration. The output of the present level is another intermediate document (the "POS+M COMBINED" document in Figure 16) containing the morphosyntactic and the morphological combined annotations for the input document.

As with morphosyntactic category and Lemma combination rules, before describing the rules used for morphological combination, it seems convenient to introduce the elements and the notation used to specify and represent them. Therefore, in Subsection 5.3.3.1, the main mathematical concepts underlying the notation are surveyed. Then, the elements in the notation of this particular type of combination rules will be described in Subsection 5.3.3.2. Finally, the rules for morphological combination will be enumerated and explained (when necessary) in Subsection 5.3.3.3.

5.3.3.1 MATHEMATICAL TERMS APPLIED IN THE NOTATION

The main mathematical terms and notions applied in the specification of the rules for morphological combination are presented in this subsection. The definitions of these terms have been taken mainly from Caballero-Roldán *et al.* (2007), as the definitions for morphosyntactic category and Lemma combination rules.

A **partial order** is a binary relation " \leq " over a set A that is reflexive, antisymmetric and transitive, *i.e.*, for all a, b and c in A , the following three conditions hold:

- $a \leq a$ (reflexivity);
- if $a \leq b$ and $b \leq a$, then $a = b$ (antisymmetry);
- if $a \leq b$ and $b \leq c$, then $a \leq c$ (transitivity).

A pair (A, \leq) , where A is a set and “ \leq ” is a partial order over A , is called a **partially ordered set** (or a **poset**).

Let (A, \leq) be a partially ordered set, and let x and y be two elements in A . An element z of A is the **join** (or *least upper bound* or **supremum**) of x and y , if the following two conditions are satisfied:

1. $x \leq z$ and $y \leq z$ (*i.e.*, z is an *upper bound* of x and y); and
2. for any w in A , such that $x \leq w$ and $y \leq w$, also $w \leq z$ (*i.e.*, z is lesser or equal to any other upper bound of x and y).

If there is a join of x and y , then it is unique, since if both z and z' are least upper bounds of x and y , then $z \leq z' \leq z$, which implies $z = z'$. If the join does exist, it is denoted by $x \vee y$.

Let (A, \leq) be a with a partially ordered set, and let x and y be two elements in A . An element z of A is the **meet** (or *greatest lower bound* or **infimum**) of x and y , if the following two conditions are satisfied:

3. $z \leq x$ and $z \leq y$ (*i.e.*, z is an *lower bound* of x and y); and
4. for any w in A , such that $w \leq x$ and $w \leq y$, also $z \leq w$ (*i.e.*, z is greater or equal to any other lower bound of x and y).

If there is a meet of x and y , then indeed it is unique, since if both z and z' are greatest lower bounds of x and y , then $z \leq z' \leq z$, which implies $z = z'$. If the join does exist, it is denoted by $x \wedge y$.

Now, let (L, \leq) be a poset. L is a **lattice** if and only if for all elements x and y of L , the set $\{x, y\}$ has both a join (or supremum) and a meet (or infimum).

Due to the necessary existence of the corresponding joins and meets in a lattice, \vee and \wedge are binary operations. Accordingly, lattices can also be characterized as algebraic structures by means of these two binary operations. Hence, as any other algebraic structure, lattices satisfy certain axiomatic identities, which are described next.

Let L be a set with two binary operations, \vee and \wedge . A *lattice* is an algebraic structure (L, \vee, \wedge) , such that the following axiomatic identities hold for all members a, b , and c of L :

Commutativity	$a \vee b = b \vee a$	$a \wedge b = b \wedge a$
Associativity	$a \vee (b \vee c) = (a \vee b) \vee c$	$a \wedge (b \wedge c) = (a \wedge b) \wedge c$
Absorption	$a \vee (a \wedge b) = a$	$a \wedge (a \vee b) = a$
Idempotence	$a \vee a = a$	$a \wedge a = a$

Given a set, S , the **power set** (or **powerset**) of S , written $\mathcal{P}(S)$, is the set of all subsets of S . It can be trivially shown that the triple $(\mathcal{P}(S), \cup, \cap)$ (or, equivalently, the pair $(\mathcal{P}(S), \subseteq)$) is a lattice for any (nonempty) set S .

5.3.3.2 DESCRIPTION OF THE NOTATION

The particularities of the notation formalism for morphological combination rules are described in this subsection. The (meta)variables, functions and procedures required for the representation of these rules are the following:

- **(Meta)Variables.**
 - $TOOLS$ is a variable that stands for the whole set of the linguistic tools integrated into the architecture configuration. In this case, $TOOLS = \{NDLPT, PLPMT, LPMT\}$.
 - SET_{TOOLS} designates the set of tools that generated a morphological annotation for a token into the configuration. Obviously, in this case, $SET_{TOOLS} \subseteq TOOLS$. If a tool has not produced an annotation for a given morphological attribute of a token, then the morphological tag assigned for that attribute of the token is the empty set (\emptyset).
 - $cat_{COMB} [offset]$ is an array that contains the contextual information about the combined morphosyntactic categories obtained in a previous combination sub-phase for the different tokens in the input document. For example, if $offset = 0$, it references the current token; if $offset = -1$, it references the previous token; and if $offset = 1$, it references the following token.
 - $lemma_{COMB} [offset]$ designates another array. It contains the contextual information about the combined lemmas obtained in a previous combination sub-phase for the different tokens in the input document. The examples for $offset$ in the previous definition hold also for this one.
 - $lemma [offset]$ is a metavariable that designates any array of lemmas used within this combination sub-phase. The examples for $offset$ in the previous definitions hold also for this one.
 - V_A refers to the set of all the possible values that the morphological attribute A can take.

- $\mathcal{P}(V_A)$ designates the set of all subsets of V_A corresponding to a given morphological attribute A ; as shown in the previous subsection, $(\mathcal{P}(V_A), \cup, \cap)$ or, equivalently, the pair $(\mathcal{P}(V_A), \subseteq)$, is a lattice for any attribute A .
 - V_A^t refers to the set of the values that a linguistic annotation tool, $t \in SET_{TOOLS}$, assigned to the morphological attribute A . Evidently, $V_A^t \in \mathcal{P}(V_A)$.
 - $V_A^t[offset]$ is an array that stores the V_A^t associated by a linguistic annotation tool t , $t \in SET_{TOOLS}$, not only to the current token but also to the previous and posterior ones. It provides the combination rules with a contextual window that can help discard and/or select some of the values assigned to the attribute A by applying grammatical agreement rules.
 - V_A^{COMB} is an output variable. Eventually, it contains the combined (set of) value(s) for the attribute A . Also in this case $V_A^{COMB} \in \mathcal{P}(V_A)$.
 - $V_A^{COMB}[offset]$ is another array, and it stores the different V_A^{COMB} , not only for the current token but also for the previous ones and posterior ones. It is used in conjunction with the array $V_A^t[offset]$ for the same purposes.
- **Functions.**
- *Is_Cat* (*lemma*, *category*, *offset*) returns TRUE when the input parameter *lemma* can be associated to the input *category* for a given token and returns FALSE otherwise. The token in question is the one referenced by the *offset* input parameter as explained above: if *offset* = 0, it references the current token; if *offset* = -1, it references the previous token; etc. For example, for the Spanish phrase “*El París del siglo XIX*” (“the Paris of the 19th century”), assuming that the current token is “*el*”,
 - $Is_Cat("el", AT, 0) = TRUE$
 - $Is_Cat("el", AJ, 0) = FALSE$
 - $Is_Cat("París", AT, -1) = TRUE$
 - *Is_Lemma*(*lemma_array*, *offset*, *string*) returns TRUE if *string* and the component *offset* of the *lemma_array* (that is, *lemma_array[offset]*) are equal, returning FALSE in other case. For example, assuming that the current token is “*París*”:
 - $Is_Lemma(["el", "París"], 0, "París") = TRUE$
 - $Is_Lemma(["el", "París"], -1, "el") = FALSE$
 - *Is_Value*(V_A , *attribute*, *value*) returns TRUE if *value* is included in V_A (that is, if $value \in V_A$), and FALSE in other case. In this function, it is assumed that the input V_A is the set of values that the input *attribute* can take (that is, $A = attribute$).

- *First_Letter* (*lemma*, *value*) returns TRUE if the first character (letter) of the input parameter *lemma* equals the input parameter *value*, returning FALSE in other case.
- Since $(\mathcal{P}(V_A), \cup, \cap)$ is a lattice for any attribute *A*, the operations meet and join can be defined for every pair of sets $V_A^i, V_A^j \in \mathcal{P}(V_A)$ as follows:

- $meet(V_A^i, V_A^j) = V_A^i \cap V_A^j$
- $join(V_A^i, V_A^j) = V_A^i \cup V_A^j$

Unitary sets of values are represented simply by the code of the value, without any further addenda; sets whose cardinality is greater than 1, following EAGLES (1996a), are notated between brackets and with one value separated from the other by the alternative symbol ‘|’. Therefore, the join of two sets of attribute values, for annotation purposes, can be obtained by the apposition of their different values by means of the alternative symbol ‘|’ (duplicates are removed, since they are not multisets but sets). For example, for the GENDER attribute:

- $meet(V_{GENDER}^i, V_{GENDER}^j) = meet(M, F) = M \cap F = \emptyset$
- $meet(V_{GENDER}^{i'}, V_{GENDER}^j) = meet([M|F], F) = [M|F] \cap F = F$
- $join(V_{GENDER}^i, V_{GENDER}^j) = join(M, F) = M \cup F = [M|F] (= \{masculine, feminine\})$
- $join(V_{GENDER}^{i'}, V_{GENDER}^j) = join([M|F], F) = [M|F] \cup F = [M|F]$
- $join(V_{GENDER}^{i'}, V_{GENDER}^{j''}) = join([M|F], N) = [M|F] \cup N = [M|F|N]$
(=*masculine, feminine, neuter*)

Due to the commutativity and associativity of these operations in the lattice $(\mathcal{P}(X), \cup, \cap)$, they can be extended as iterated binary operations and be performed over a whole family of *n* attribute value sets (not just two of them):

- $meet(V_A^1, \dots, V_A^n) = V_A^1 \cap \dots \cap V_A^n$
- $join(V_A^1, \dots, V_A^n) = V_A^1 \cup \dots \cup V_A^n$

▪ **Procedures.**

- *Assign*(V_A^1, V_A^2) copies the set of values stored in V_A^2 to V_A^1 .
- *Assign_Attrib*($V_A^1[*offset*], V_A^2[*offset*], *attribute*) copies the set of values associated to the input parameter *attribute*, $V_A^2[*offset*]$, to the *offset* component of the array of sets V_A^1 .$
- *Assign_Attrib_Value*($V_A[*offset*], *attribute*, *value*) assigns the input *value*, suitable for the input *attribute*, to the *offset* component of the array of attribute values V_A .$

- *Assign_Category*($cat_{COMB}[offset]$, *category*) assigns the input *category* to the *offset* component of the array of combined categories cat_{COMB} .

5.3.3.3 MORPHOLOGICAL COMBINATION RULES

The rules applied within the morphological combination sub-phase of OntoTagger are presented in this subsection. They are used in the comparison of the different values assigned to each particular morphological attribute assigned to each token by the different linguistic tools integrated into the system. These values are compared in order to select the most accurate(s) of them for the token and the attribute in question. The selected value(s) constitute(s) eventually the combined morphological annotation attached by OntoTagger to this token for that particular attribute.

This combination sub-phase goes through two different stages, namely (i) the selection of the candidate values for each attribute of a given token; and (ii) the distilment of the set of values selected into a final, combined tag as precise as possible from the information available.

Each of these two stages has been encapsulated into a Java function, respectively: *MorphologicalValuesSelection* and *MorphologicalRefinement* (described below). Whereas the former formalises common sense knowledge, the second one formalises linguistic (grammatical) knowledge, dealing mainly with Spanish agreement rules or with closed class categories (that is, non-lexical) word forms. Each rule in these packages is accompanied with a simple explanation of its meaning, describing the knowledge it tries to capture.

Each of these Java functions include a particular package of rules, where only one of them is to be triggered and fired at a time for a given attribute of a token, after applying the selection criteria mentioned at the beginning of Section 5.3. After a rule is fired, the working memory of the package (*i.e.*, at least one of its internal variables) will be updated. This might cause (an)other rule(s) in the package to trigger and fire afterwards. The sub-phase ends when there remains no rule triggered.

In what follows, we assume that, in each rule, the set V_A^{COMB} is always initialised with the empty set (\emptyset) before any *meet* or *join* operation is applied to it. Besides, if a tool has not assigned a morphological value tag to a token, then it is assigned an empty value tag (\emptyset).

- **First stage – *MorphologicalValuesSelection*.** As mentioned above, this stage deals with the selection of the candidate values for each attribute of a given token, referred to as the current token. Initially, for a particular attribute of the current token, there are as many sets of values as linguistic annotations tool integrated into OntoTagger. Each one of these sets of values comes from one of the linguistic tools. The resulting candidate values are obtained (a) discarding from the input sets of

values the ones that cannot be ascribed to the attribute considered (for the current token); and (b) joining the ones left in these input sets. The package of rules that formalises this first stage is presented next.

- **RULE #1:** $|SET_{TOOLS}| = n \wedge \forall V_A^1, \dots, V_A^n [V_A^1 = \dots = V_A^n] \rightarrow Assign(V_A^{COMB}, V_A^1)$
 - **Explanation:** When all the linguistic annotation tools agree on the set of values V_A^i that can be ascribed to an attribute A of a token, then the set of values selected in this stage is obtained by simply copying it from any of the morphological value sets obtained coming from the linguistic tools.
- **RULE #2:** $|SET_{TOOLS}| = n \wedge \exists V_A^1, \dots, V_A^n / meet(V_A^1, \dots, V_A^n) = \emptyset \rightarrow Assign(V_A^{COMB}, join(V_A^1, \dots, V_A^n))$
 - **Explanation:** When the value sets (V_A^i) for a morphological attribute A coming from linguistic tools ($i = 1, \dots, n$) do not share any member (or, equivalently, when their intersection and, therefore, their *meet* equals the empty set, \emptyset), then none of their members can be assumed to be wrong values for the attribute and discarded. Accordingly, all of them are candidates that have to be considered in the following stage, and the combined set of values of this stage is calculated as the union (*join*) of all these sets.
- **SPECIAL CASE #2.1:** $\exists V_A^i, V_A^j \in \mathcal{P}(V_A) [V_A^i = \emptyset \wedge V_A^j \neq \emptyset] \rightarrow Assign(V_A^{COMB}, V_A^j)$
 - **Explanation:** If i is a linguistic tool that has not determined a value for the attribute A , (and, therefore, its associated value set, V_A^i , equals the empty set, \emptyset), but there is another tool, j , whose associated value set, V_A^j , is not the empty set, then the temporary combined set of values of this stage can be obtained by simply assigning the values of the nonempty set of values to the combination set of morphological values.
- **RULE #3:** $\exists V_A^1, \dots, V_A^n \in \mathcal{P}(V_A) / meet(V_A^1, \dots, V_A^n) \neq \emptyset \rightarrow Assign(V_A^{COMB}, meet(V_A^1, \dots, V_A^n))$
 - **Explanation:** When the value sets (V_A^i) for a morphological attribute A coming from linguistic tools ($i = 1, \dots, n$) share some member(s) (or, equivalently, when their intersection and, therefore, their *meet* is not the empty set, \emptyset), then all these shared values are candidates to be correct values for the attribute. Therefore, the combined set of values of this stage can be obtained simply as the intersection (*meet*) of all these sets.
- **Second stage – MorphologicalRefinement.** As stated above, the set of the morphological values selected in the previous stage has to be distilled into a combined tag as precise as possible from the information available. This distillation involves reducing the cardinality of the selected set of morphological values to 1 or else at a minimum. The resulting distilled tag is attached by

OntoTagger to the current token as its combined morphological annotation for the attribute in hand. The package of rules that formalises this second stage is presented next.

- **RULE #4:** $[Is_Cat(lemma_{COMB}, NP, 0) \vee Is_Cat(lemma_{COMB}, NC, 0) \vee Is_Cat(lemma_{COMB}, AJ, 0)] \wedge Is_Cat(lemma_{COMB}, AT, -1) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], NUMBER)$
 - **Explanation:** In Roman languages, the articles, the adjectives and the nouns (either proper or common) agree in Gender and Number in a Nominal Phrase (NP). Therefore, whenever an Article precedes an Adjective or a Noun in an NP, and the Gender and/or the Number of the Article (easier to determine and/or disambiguate) are (is) known, then they are (it is) broadcasted to the Noun or the Adjective following it.
- **RULE #5:** $Is_Cat(lemma_{COMB}, NP, 0) \wedge [Is_Cat(lemma_{COMB}, PU, 1) \vee (Is_Cat(lemma_{COMB}, VM, 1) \wedge Is_Lemma(lemma_{COMB}[1], "ser"))] \wedge Is_Cat(lemma_{COMB}, AT, 2) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[2], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[2], NUMBER)$
 - **Explanation (part 1):** In a sequence such as <Proper Noun, Punctuation Mark, Article>, the Article most likely refers to the Proper Noun (at least in Spanish) and therefore, they agree in Gender and Number (except for metaphoric uses). In this cases, if the Gender and/or the Number of the Article (easier to determine and/or disambiguate) are (is) known, then they are (it is) broadcasted to the preceding Proper Noun.
 - **Explanation (part 2):** In Spanish copulative sentences, the Subject and the (nominal) Predicate usually agree in Gender and Number (except for metaphoric uses) for coherence reasons (this is a Discourse Level heuristic). Therefore, whenever a Copula stands in-between a Proper Noun (preceding) and an Article (postponed), and the Gender and/or the Number of the Article (easier to determine and/or disambiguate) are (is) known, then they are (it is) broadcasted to the Proper Noun preceding the Copula.
- **RULE #6:** $Is_Cat(lemma_{COMB}, AJ, -1) \wedge [Is_Cat(lemma_{COMB}, NP, 0) \vee Is_Cat(lemma_{COMB}, NC, 0)] \wedge meet(V_A[-1], V_A[0]) \neq V_A[0] \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], NUMBER)$
 - **Explanation:** In Roman languages, adjectives and nouns (either proper or common) agree in Gender and Number in a Nominal Phrase (NP). Therefore, whenever an Adjective precedes a Noun in an NP, and the Gender and/or the Number of the Noun are (is) known and does not coincide with the one(s) of the Adjective, then they are (it is) broadcasted from the Noun to the Adjective preceding it.

- **RULE #7:** $Is_Cat(lemma_{COMB}, NC, -1) \wedge Is_Cat(lemma_{COMB}, AJ, 0) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], GENDER)$
 - **Explanation:** Similar to the previous rule, but in this case the Adjective is postponed.
- **RULE #8:** $Is_Cat(lemma_{COMB}, AJ, -1) \wedge Is_Cat(lemma_{COMB}, PDL P, 0) \wedge Is_Cat(lemma_{COMB}, V, 1) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], NUMBER)$
 - **Explanation:** This is an empirically-determined rule and is applied to the Spanish sequences <Adjective, Relative Pronoun, Verb>, where the Relative Pronoun is usually ‘que’. The Gender and the Number of this Relative Pronoun are most difficult to determine without the help of some contextual knowledge. This rule permits deriving the Gender and the Number of this Relative Pronoun from the Gender and the Number of the Adjective, basically using Spanish agreement rules.
- **RULE #9:** $Is_Cat(lemma_{COMB}, NC, -1) \wedge Is_Cat(lemma_{COMB}, PDL P, 0) \wedge Is_Cat(lemma_{COMB}, V, 1) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], NUMBER)$
 - **Explanation:** Similar to RULE #8, but for the sequences following the pattern <Common Noun, Relative Pronoun, Verb>.
- **RULE #10:** $Is_Cat(lemma_{COMB}, ATD, -1) \wedge Is_Cat(lemma_{COMB}, PDL P, 0) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], NUMBER)$
 - **Explanation:** Similar to RULE #8, but for the sequences following the pattern <Definite Article, Relative Pronoun>.
- **RULE #11:** $Is_Cat(lemma_{COMB}, AJ, 0) \wedge Is_Cat(lemma_{COMB}, NC, 1) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER)$
 - **Explanation:** Similar to the RULE #7, but in this case the Adjective precedes the Noun.
- **RULE #12.1:** $Is_Cat(lemma_{COMB}, VM, -1) \wedge Is_Cat(lemma_{COMB}, PDL P, 0) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], NUMBER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], CASE, "C") \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], PERSON) \wedge Assign_Attrib_Value(V_A^{COMB}[-1], VOICE, "A") \wedge Assign_Attrib_Value(V_A^{COMB}[-1], REFLEXIVITY, "R")$
 - **Explanation:** In Spanish, a Main Verb followed by a Reflexive Pronoun identifies a Reflexive verbal form. Therefore, the Voice of the Verb must be Active and its Reflexivity attribute is set to Reflexive; and the Reflexive

Pronoun inherits the values of Number and Person from the Verb and, also, the value Accusative can be assigned to its Case attribute.

- **RULE #12.2:** $Is_Cat(lemma_{COMB}, PDFP, 0) \wedge Is_Cat(lemma_{COMB}, VA, 1) \wedge Is_Cat(lemma_{COMB}, VM, 2) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], CASE, "C") \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], PERSON) \wedge Assign_Attrib_Value(V_A^{COMB}[1], VOICE, "A") \wedge Assign_Attrib_Value(V_A^{COMB}[1], REFLEXIVITY, "R")$
 - **Explanation:** In Spanish, the sequence <Reflexive Pronoun, Auxiliary Verb, Main Verb> might reveal either a Mediopassive (passive-reflexive) Voice (if the Person verbal attribute is Third) or a Reflexive form of the Main Verb. Whereas no evidence supporting the latter was found in the corpus used in the development of OntoTagger, some evidence supporting the former was indeed found, and this is what this rule formalises. The attribute value assignment is analogous to the one in RULE #12.1.
- **RULE #12.3:** $Is_Cat(lemma_{COMB}, PDFP, 0) \wedge Is_Cat(lemma_{COMB}, VM, 1) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], CASE, "C") \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], PERSON) \wedge Assign_Attrib_Value(V_A^{COMB}[1], VOICE, "A") \wedge Assign_Attrib_Value(V_A^{COMB}[1], REFLEXIVITY, "R")$
 - **Explanation:** Similar to RULE #12.2, but for the sequence <Reflexive Pronoun, Main Verb>.
- **RULE #13:** $Is_Cat(lemma_{COMB}, PD, -1) \wedge Is_Cat(lemma_{COMB}, AJ, 0) \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[-1], GENDER)$
 - **Explanation:** Similar to RULE #7, but for the sequence <Pronoun/Determiner, Adjective>.
- **RULE #14.1:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, NC, 1) \wedge First_Letter(lemma_{COMB}, 'm') \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PERSON, 1) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PD_POSSESSOR_NUMBER, "S")$
 - **Explanation 1:** In Spanish, a Possessive Pronoun/Determiner starting with an 'm' reveals a Singular Number Of Possessors and a First Person, and these are two of the values assigned.
 - **Explanation 2:** According to linguistic agreement rules, a Pronoun/Determiner agrees in Gender and Number with the Noun they qualify or stand for; therefore, in this

rule, the Possessive Pronoun/Determiner inherits the Number and Gender values from the postponed Noun.

- **RULE #14.2:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, NC, 1) \wedge First_Letter(lemma_{COMB}, 't') \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PERSON, 2) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PD_POSSESSOR_NUMBER, "S")$
 - **Explanation 1:** In Spanish, a Possessive Pronoun/Determiner starting with a 't' reveals a Singular Number Of Possessors and a Second Person, and these are two of the values assigned.
 - **Explanation 2:** See Explanation 2 in RULE #14.1.
- **RULE #14.3:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, NC, 1) \wedge First_Letter(lemma_{COMB}, 's') \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PERSON, 3)$
 - **Explanation 1:** In Spanish, a Possessive Pronoun/Determiner starting with an 's' reveals a Third Person, and this is one of the values assigned.
 - **Explanation 2:** See Explanation 2 in RULE #14.1.
- **RULE #14.4:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, NC, 1) \wedge First_Letter(lemma_{COMB}, 'n') \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PERSON, 1) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PD_POSSESSOR_NUMBER, "P")$
 - **Explanation 1:** In Spanish, a Possessive Pronoun/Determiner starting with an 'n' reveals a Plural Number Of Possessors and a First Person, and these are two of the values assigned.
 - **Explanation 2:** See Explanation 2 in RULE #14.1.
- **RULE #14.5:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, NC, 1) \wedge First_Letter(lemma_{COMB}, 'v') \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PERSON, 2) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PD_POSSESSOR_NUMBER, "P")$

- **Explanation 1:** In Spanish, a Possessive Pronoun/Determiner starting with a ‘v’ reveals a Plural Number Of Possessors and a Second Person, and these are two of the values assigned.
- **Explanation 2:** See Explanation 2 in RULE #14.1.
- **RULE #15.1:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, AJ, 1) \wedge First_Letter(lemma_{COMB}, 'm') \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PERSON, 1) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PD_POSSESSOR_NUMBER, "S")$
 - **Explanation 1:** See Explanation 1 in RULE #14.1.
 - **Explanation 2:** According to linguistic agreement rules, a Possessive Pronoun/Determiner agrees in Gender and Number with an Adjective co-occurring in the same NP; therefore, by means of this rule, the Possessive Pronoun/Determiner inherits the Number and Gender values from the postponed Adjective.
- **RULE #15.2:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, AJ, 1) \wedge First_Letter(lemma_{COMB}, 't') \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PERSON, 2) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PD_POSSESSOR_NUMBER, "S")$
 - **Explanation 1:** See Explanation 1 in RULE #14.2.
 - **Explanation 2:** See Explanation 2 in RULE #15.1.
- **RULE #15.3:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, AJ, 1) \wedge First_Letter(lemma_{COMB}, 's') \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge Assign_Attrib_Value(V_A^{COMB}[0], PERSON, 3)$
 - **Explanation 1:** See Explanation 1 in RULE #14.3.
 - **Explanation 2:** See Explanation 2 in RULE #15.1.
- **RULE #15.4:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, AJ, 1) \wedge First_Letter(lemma_{COMB}, 'n') \rightarrow Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib(V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge$

$Assign_Attrib_Value (V_A^{COMB}[0], PERSON, 1) \wedge Assign_Attrib_Value (V_A^{COMB}[0], PD_POSSESSOR_NUMBER, "P")$

- **Explanation 1:** See Explanation 1 in RULE #14.4.

- **Explanation 2:** See Explanation 2 in RULE #15.1.

- **RULE #15.5:** $[Is_Cat(lemma_{COMB}, PDOP, 0) \vee Is_Cat(lemma_{COMB}, PDOD, 0)] \wedge Is_Cat(lemma_{COMB}, AJ, 1) \wedge First_Letter(lemma_{COMB}, 'v') \rightarrow Assign_Attrib (V_A^{COMB}[0], V_A^{COMB}[1], GENDER) \wedge Assign_Attrib (V_A^{COMB}[0], V_A^{COMB}[1], NUMBER) \wedge Assign_Attrib_Value (V_A^{COMB}[0], PERSON, 2) \wedge Assign_Attrib_Value (V_A^{COMB}[0], PD_POSSESSOR_NUMBER, "P")$

- **Explanation 1:** See Explanation 1 in RULE #14.5.

- **Explanation 2:** See Explanation 2 in RULE #15.1.

- **RULE #16:** $Is_Cat(lemma_{COMB}, VA, 0) \wedge Is_Cat(lemma_{COMB}, VMP, 1) \wedge Is_Lemma(lemma_{COMB} [0], "ser") \rightarrow Assign_Attrib_Value (V_A^{COMB}[0], VOICE, "P") \wedge Assign_Attrib_Value (V_A^{COMB}[0], REFLEXIVITY, "N")$

- **Explanation:** The Auxiliary Verb ‘ser’ in Spanish (*i.e.*, ‘être’ in French, ‘essere’ in Italian, ‘to be’ in English, ‘werden’ in German, etc.), followed by a Predicative Main Verb, identifies a Passive verbal form. Therefore, the Voice attribute of the Verb is assigned the value Passive and its Reflexivity attribute is set to Irreflexive.

- **RULE #17:** $[Is_Cat(lemma_{COMB}, NC, 0) \vee Is_Cat(lemma_{COMB}, NP, 0)] \wedge Is_Cat(lemma_{COMB}, PU[09/10/13], 1) \wedge Is_Cat(lemma_{COMB}, NC, 2) \wedge Is_Lemma(lemma_{COMB} [2], "s") \rightarrow Assign_Attrib_Value (V_A^{COMB}[0], CASE, "G") \wedge Assign_Category (cat_{COMB} [1], "PU13")$

- **Explanation:** This is an English-related rule covering the particular syntax of Saxon Genitive: the sequence (noun –either common or proper–, “ ’ ”, “s”) reveals a Saxon Genitive form; therefore, the Punctuation Mark can be undoubtedly and unambiguously be tagged as an Apostrophe (PU13) and the Case value of the Noun set to Genitive.

All the rules applied in the morphological combination sub-phase of OntoTagger have been presented up to this point. As explained, they are used in the combination of every attribute value of every token in the input document.

The morphological combination sub-phase completes the combination of annotations at the MorphoSyntactic Level within OntoTagger. So, to conclude, a set of intermediate annotated

documents are generated all along these morphosyntactic combination sub-phases: (i) a document containing the standardised and combined syntactic annotations for the input document (the “Syn COMBINED” document in the architecture configuration of Figure 16 – see the beginning of the present chapter); (ii) another one with its standardised and combined Lemma and morphosyntactic annotations (the “L+POS COMBINED” document in the aforementioned Figure 16); and (iii) another one with its standardised and combined morphosyntactic and morphological annotations (the “POS+M COMBINED” document of Figure 16).

The sub-phase of combination in OntoTagger is completed when the annotations at the `Semantic Level` have been combined as well. They are included in a new document, referred to as “Sem COMBINED” in Figure 16, but the `Semantic Level` is so complex *per se* that it requires a whole dedicated section in the present chapter for its description. This description is given in the next section, which includes its combination heuristics in a particular subsection.

5.4. THE SEMANTIC ANNOTATION MANAGER MODULE (SAMM) IN DETAIL

As stated in Section 5.2, the SAMM is a particular module that was included in OntoTagger mainly to (1) merge (that is, combine and integrate) the semantic annotations coming from the linguistic annotation tools integrated into the system; (2) merge the resulting semantic annotations with other semantic annotations generated *ad hoc*; and (3) apply afterwards these semantic and ontology-based annotations to learning, so as to evaluate their quality and properties.

As for the first goal of this system, there is not much to be merged, since the semantic annotations coming from the linguistic annotation tools are actually minimal. In effect, none of the tools integrated into OntoTagger performs real semantic annotation. Yet, they provide some semantic information that needs to be merged with the rest of the annotations already processed. Firstly, as already mentioned, LACELL’s POS tagger attaches a sort of semantic tag to adverbs, according to their traditional semantic classification in Spanish grammars (*i.e.*, the kind of circumstance they designate: location, either spatial or temporal, manner, etc.). And secondly, Connexor’s FDG Parser labels a number of tokens with some syntactic dependencies (time, duration, frequency, quantity, manner, location, source, goal, etc.) that can also be considered semantic as well. Both of these two sets of tags and labels can be considered a type of role labelling for the tokens in question, and they have already been conveniently standardised (before being inputted to this module) by mapping them onto OntoTag’s ontologies in the standardisation phase. Accordingly, they just have to be integrated with the

remaining ones in this module. Therefore, the layer of role labelling (at least for circumstances) had to be implemented in OntoTagger¹⁹⁰.

The second goal of this system (*i.e.*, merging these semantic annotations with other ontology-based annotations) tried to (i) extend the layer of sense tagging to other lexical units (apart from adverbs); and (ii) include the annotation of the particular named entities of the domain.

In order for sub-goal (i) to be attained, this layer has been divided into two different sub-layers, each one corresponding to a different type of sense tagging. On the one hand, it includes a sub-layer for domain independent sense tagging. In this sub-layer, each content word found in the input document is attached a EuroWordNet-based annotation¹⁹¹. First, the SAMM looks up the content word in EuroWordNet and retrieves the identifiers of the synsets into which this word is included. Second, these identifiers are included in a list. Finally, this list is attached to the content word in hand as its sense tag, without any further processing. Obviously, this list is not necessarily unitary and, therefore, it constitutes in most cases a non-disambiguated sense tagging of content words¹⁹². On the other hand, this layer includes also a sub-layer for domain dependent sense tagging. In this other sub-layer, each word that is a particular term in the domain of the texts being annotated is assigned an additional sense tag. This additional sense tag is a reference to its corresponding concept of the Cinema Named Entities Ontology (CNEO – a domain ontology elaborated for this purpose). These two forms of sense tagging constitute the Sense Tagging Layer of OntoTagger.

In order for sub-goal (ii) –that is, named entity annotation– to be attained, this module includes an expressly developed sub-module for the recognition, (sub)classification and tagging of named entities, according to the MUC and ACE tagsets (further specified by means of the CNEO) and, therefore, it implements the Instance Semantic Annotation Layer of OntoTagger. This additional semantic annotation layer has to be (intra-level) merged by this module with the annotations obtained in the Sense Tagging Layer and in the Semantic Role Labelling Layer (commented above).

Lastly, the third goal of this system (*i.e.*, learning) basically consists of (i) populating the domain ontology with the instances identified within the process of named entity recognition and (sub)classification; and (ii) updating, when appropriate, the semantic lexicon and the gazetteers included in this configuration with the semantic information extracted by the rest of the processes of this module.

¹⁹⁰ For the sake of time and human resources, the annotations of FDG considered semantic were decanted into the syntactic level and were not combined with the minimal semantic annotations coming from the LACELL's POS tagger. This allowed incorporating a full processing of named entities instead.

¹⁹¹ EuroWordNet is considered a linguistic ontology by Gómez-Pérez *et al.* (2004), amongst other authors.

¹⁹² Word Sense Disambiguation of these lists is to be tackled in further experiments with OntoTagger, but it has already been enabled by the inclusion of the whole set of synsets corresponding to content words in this implementation of OntoTag's architecture.

Thus, briefly, the SAAM can be regarded not only as a type of annotation process but also as a sort of linguistic knowledge-based machine learning process as well. The architecture of the SMM is sketched in Figure 19 (its contextual diagram, with its inputs and its output) and in Figure 20 (a sort of first-level process diagram of the module, with its main internal processes).

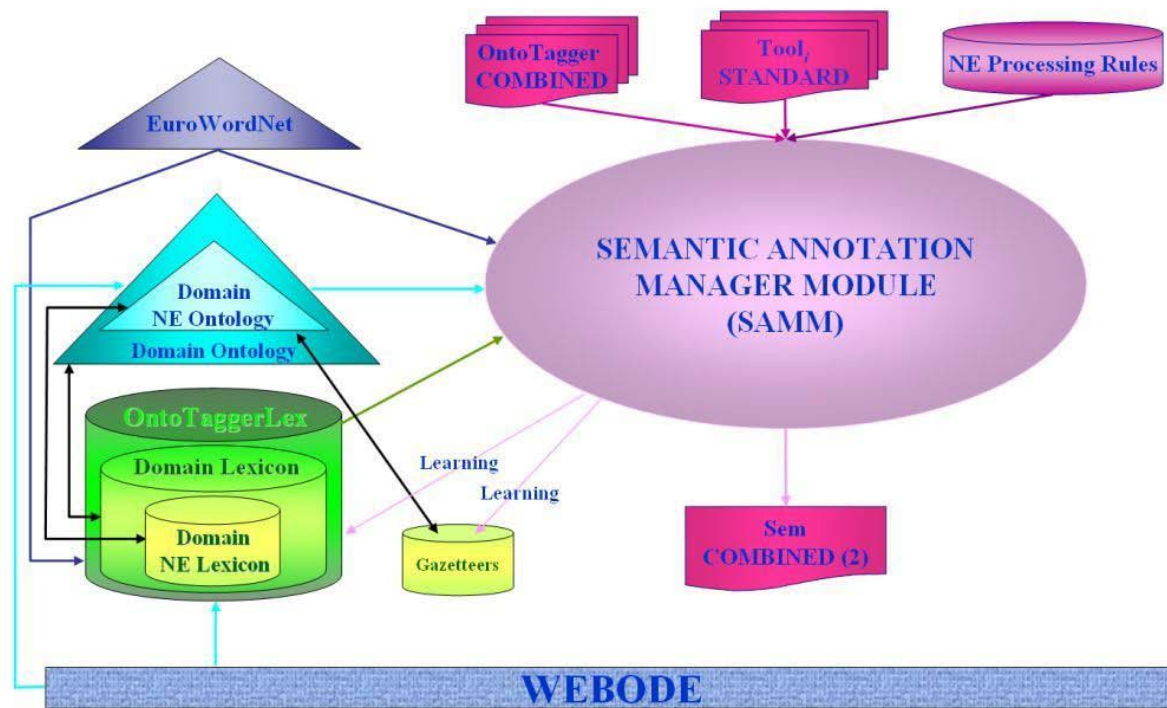


Figure 19: Semantic Annotation Manager Module – contextual diagram: inputs and output

As can be observed in Figure 19, the inputs to the SMM are multiple and of a variety of kinds. The main inputs are, as usual, the annotation documents generated within the system. On the one hand, this module uses the files with the separate annotations (up to the Semantic Level) of the linguistic tools integrated into OntoTagger, conveniently decanted and standardised (the “Tool; STANDARD” files in Figure 19). On the other hand, it also needs the combined annotations already generated in the other combination sub-phases of OntoTagger. Hence, this additional set contains (a) the documents with the combined annotations associated to the lower levels, that is, the Lemma, the morphosyntactic category, the morphological and the syntactic combined annotations; together with (b) the document with the minimal semantic annotations coming from the linguistic tools. Accordingly, this additional set of input files available consists of the “L+POS COMBINED”, the “L+M COMBINED” and the “Syn COMBINED” files in Figure 16 and the “OntoTagger COMBINED” files in Figure 19¹⁹³.

¹⁹³ The remaining COMBINED files in Figure 16 (that is, the “Sem COMBINED (1)” and the “Sem COMBINED (2)” files, are generated within this module of OntoTagger.

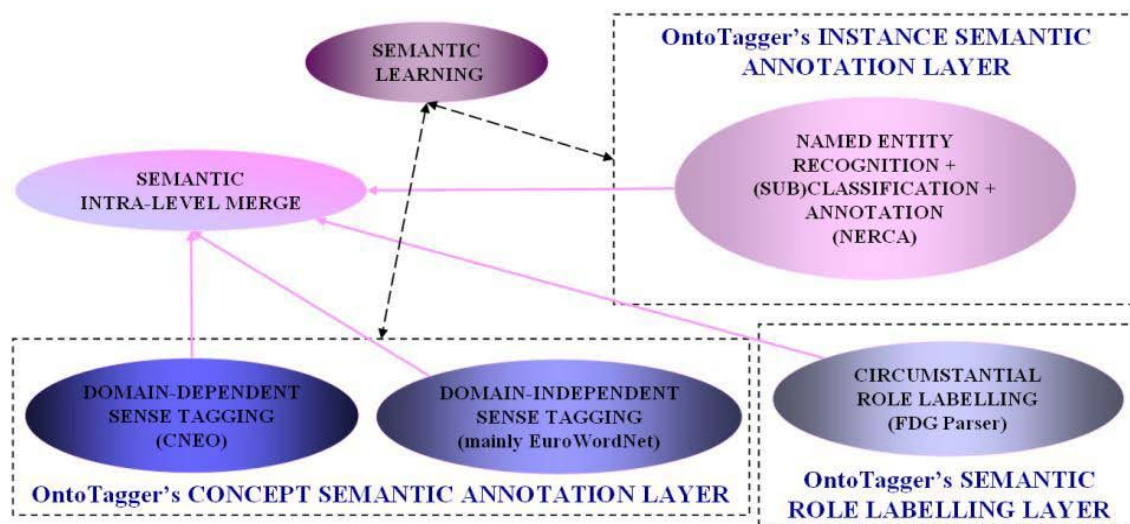


Figure 20: Semantic Annotation Manager Module in detail: internal tasks

The second important set of inputs to the SAMM is the two ontological resources integrated into this module of OntoTag’s architecture configuration, namely EuroWordNet and the CNEO. First, EuroWordNet is used mainly for sense-tagging the content words in the input document, as explained above¹⁹⁴. Second, the CNEO (Cinema Named Entity Ontology) domain ontology initially contained the concepts associated to the domain of interest, which are used for sense-tagging the input document. By means of the semantic learning process of this module, it was further populated in order to include some domain instances too. These learned instances are used as a sort of domain named entity cache in posterior named entity annotations performed by OntoTagger.

The third crucial input of the SAMM is OntoTaggerLex, the computational semantic lexicon developed specifically for this prototype. It includes a dedicated sub-lexicon for the cinema domain in which cinema named entities play a crucial role, as can be seen in Figure 19. Each time a named entity is recognised in the input file and annotated, it is stored as an instance of its corresponding concept in the CNEO. However, the CNEO cannot store all the linguistic information accompanying these named entities as annotation, provided that it is not a linguistic ontology but a domain ontology. Thus, they and their corresponding linguistic information must be stored somewhere else as well, *i.e.*, in OntoTaggerLex. Therefore, the entries for these named entities in OntoTaggerLex include all the linguistic information that cannot be stored in the CNEO, and both the CNEO and OntoTaggerLex must cooperate in the annotation of named entities within OntoTagger. OntoTaggerLex is discussed in detail in a dedicated Subsection (5.4.1, page 262).

¹⁹⁴ It also contains a small set of named entities (for example, locations such as Paris), which can be used in the Instance Semantic Annotation Layer.

The final crucial input of the SAMM is the set of heuristics used for the treatment of named entities (the “NE Processing Rules” in Figure 23), which articulate and coordinate the processing of all of the knowledge carried by the rest of inputs to the NERCA subsystem. These rules are presented in detail in Section 5.4.2.1.

A last and rather secondary input of the SAMM is the set of gazetteers compiled for their use within OntoTagger. They include lists of countries and locations, film actors’ and directors’ names, etc. These gazetteers serve as a bootstrapping source of knowledge for the detection and learning of named entities within OntoTagger.

In turn, the output of the SAMM are two new combined documents for the `Semantic Level` (the “Sem COMBINED (2)” documents), as can be observed in Figure 16 and in Figure 19. This new document contains all the annotations generated by the different resources aforementioned, conveniently combined and integrated, according to their respective (sub-)layer.

As for the combination of these different types of annotations within the SAMM, as shown in Figure 20 (page 260), they have been articulated taking the three main processes related to named entities as a basis. These three processes are the `Named Entity Recognition` (NER) process, the `Named Entity (Sub)Classification` (NEC) process and the `Named Entity Annotation` (NEA) process. All of them, altogether, constitute the NERCA subsystem (`Named Entity Recognition`, `(Sub)Classification` and `Annotation`). This subsystem (1) implements the Instance Semantic Annotation Layer within OntoTagger and (2) has to decide when the other semantic annotation processes in Figure 20 must be executed and call them.

Issue (1) is described in detail in Subsection 5.4.2; concerning issue (2), first, the `Domain-Dependent Sense Tagging` (DDST) process is called whenever a candidate domain concept is found in the input document. Second, the `Domain-Independent Sense Tagging` (DIST) process is called whenever a content word is found. In addition, the Semantic Learning (SL) process is called whenever a named entity or another piece of semantic information is found, to store conveniently this piece of semantic information in the domain ontology and/or the semantic lexicon (and to update the corresponding gazetteer).

Finally, all these semantic annotations have to be integrated internally, before being merged with the annotations of the remaining levels. This is precisely the moment to call the `Semantic Intra-Level Merger` (SILM). First, it puts together the annotations of both the Concept Semantic Annotation Layer and the Instance Semantic Annotation Layer. Whereas the former semantic annotations come from the DDST and the DIST processes, the latter come from the NERCA subsystem. And second, it incorporates into the integrated annotations of the `Semantic Level` those corresponding to the

Semantic Role Labelling Layer. As commented above, these annotations come from the linguistic annotation tools included in OntoTagger (mainly from Connexor's FDG Parser, but also from LACELL's POS tagger) and are collected by the Circumstantial Role Labelling process in Figure 20.

These processes and resources, included in OntoTagger for the semantic annotation of the input documents, are discussed in more detail in the following subsections. The logical and physical designs of OntoTaggerLex are shown next, in Subsection 5.4.1. Then (Subsection 5.4.2), the NERCA subsystem and the Instance Semantic Annotation Layer of OntoTagger will be extensively described. The Concept Semantic Annotation Layer and the DDST and the DIST processes will be presented afterwards, in Subsection 5.4.3. The SILM and the Circumstantial Role Labelling processes will be discussed after that (Subsection 5.4.4). Finally, the Semantic Learning Process will be described in Subsection 5.4.5.

5.4.1. ONTOTAGGERLEX – THE SEMANTIC LEXICON

After presenting briefly the inputs, the output and the processes of the Semantic Annotation Manager Module, the design details of OntoTaggerLex are presented in this section. As commented above, OntoTaggerLex is a (computational) semantic lexicon built specifically within OntoTagger for storing linguistic information useful for semantic annotation. This linguistic information concerns the named entities learned by the SAMM and kept in the CNEO. In particular, since the CNEO is a domain ontology, it cannot store the lexical information associated to named entities. Accordingly, this lexical information is stored in OntoTaggerLex instead. The aspects relating its design are described in the present subsection.

The basis for the design and development of OntoTaggerLex were the EAGLES (1999) preliminary recommendations on lexical semantic encoding. These recommendations had to be extended at some points and updated at some others to implement a computational semantic lexicon suitable for the types of semantic annotation mentioned above. For example, it was generalised to include some information about the functional properties of lexical entries, according to the Functional Grammar of Halliday (1994;1996).

In what follows, it is explained how these recommendations were extended and adapted when designing OntoTaggerLex. The eventual logical design of OntoTagger has been included in Figure 21 (page 264), where its Entity-Relationship (ER) Model is shown. Its related physical design diagram is included in Figure 22 (page 265).

One of the main contributions of OntoTaggerLex is its re-definition of the 'word sense identifier' attribute. This attribute was defined in EAGLES (1999) as an INTEGER that referred to a WordNet

3.5.2 synset. Obviously, this definition depended too much on this lexical resource, and had to be generalised, so that the senses of lexical entries could be specified with reference to other lexical resources, such as lexical databases, electronic dictionaries, glossaries, or a domain ontology. Thus, the EAGLES (1999) ‘word sense identifier’ attribute was re-defined as follows:

```
word-sense-id →
  [ - TERM-ID :    INTEGER
    - RESOURCE :   (WordNet | EuroWordNet | EAGLES | SIMPLE |
                    HallidayOnto | GUME | CNEO)
  ]
```

where the meaning of each element can be further explained in this way:

1. **TERM-ID:** This is an `INTEGER` value that refers to the concept identifier (or the synset identifier, when as with WordNet or EuroWordNet) that conceptualises the sense of the entry in a given ontology or lexical resource, which is identified in the following attribute.
2. **RESOURCE:** This attribute identifies the ontological or lexical (re)source applied for the description of the entry sense, whose identifier is specified in the aforementioned attribute *TERM-ID*. The different resources integrated in OntoTagger that might serve as a reference for sense identification (and annotation) are:
 - 2.1. *WordNet*: In conjunction with the attribute `TERM-ID`, this identifier is used to refer to the WordNet synsets proposed originally within the EAGLES (1999) recommendations for identifying the sense of a lexical entry.
 - 2.2. *EuroWordNet*: In conjunction with the attribute `TERM-ID`, this identifier is used to refer to EuroWordNet synsets instead of the proposed WordNet synsets.
 - 2.3. *EAGLES*: This identifier designates the subject domains proposed within the EAGLES (1999) preliminary recommendations on lexical semantic encoding themselves (which, further specialised, could constitute an ontology for semantic field-tagging the input corpus).
 - 2.4. *SIMPLE*: This item references the SIMPLE General Ontology developed within the SIMPLE (2000) project; only its entity and property counterparts have been considered of interest for this purpose and included (after a slight adaptation) into OntoTag’s Linguistic Unit Ontology (LUO).
 - 2.5. *HallidayOnto*: Halliday’s (1985) theory of Functional Grammar was found to be a very appropriate basis for a hierarchical description of verb and circumstance types, complementary to the taxonomy of entities and properties underlying the SIMPLE General Ontology. Therefore, this theory has been adapted and conceptualised into OntoTag’s LUO as well, as a right and proper way of subspecifying the meaning (*i.e.*, the sense) of verbs and circumstances.

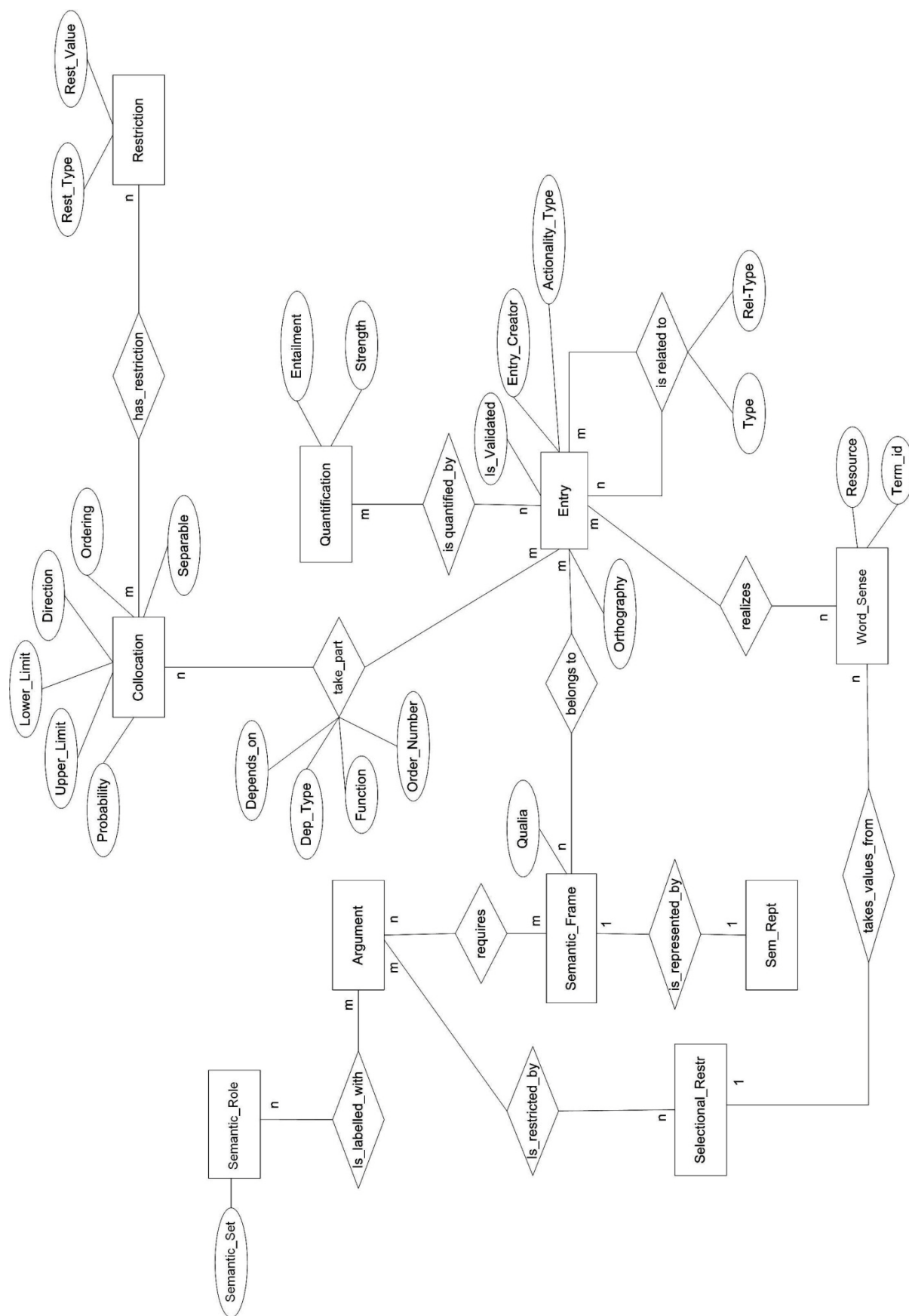


Figure 21: Entity-Relationship model of the semantic lexicon OntoTaggerLex

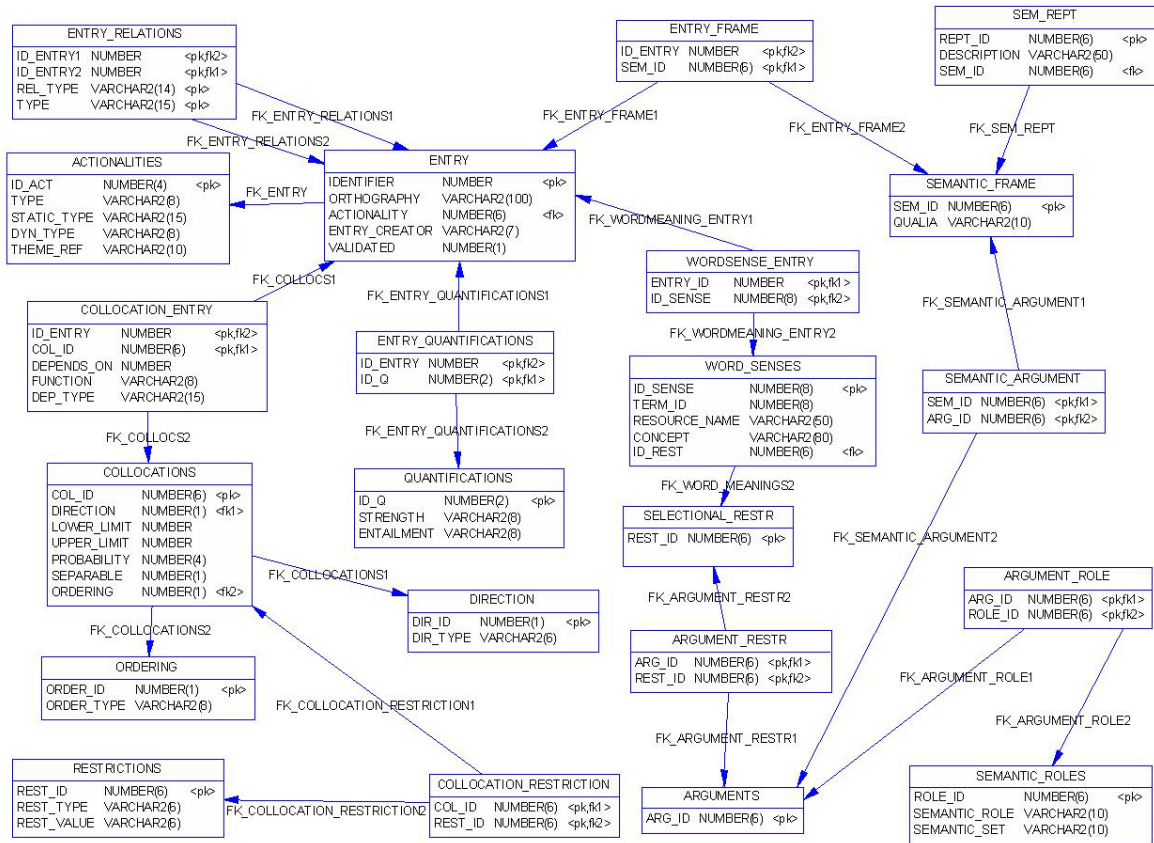


Figure 22: Physical diagram of the semantic lexicon OntoTaggerLex

- 2.6. *SIMPLE*: This item references the SIMPLE General Ontology developed within the SIMPLE (2000) project; only its entity and property counterparts have been considered of interest for this purpose and included (after a slight adaptation) into OntoTag's Linguistic Unit Ontology (LUO).
- 2.7. *HallidayOnto*: Halliday's (1985) theory of Functional Grammar was found to be a very appropriate basis for a hierarchical description of verb and circumstance types, complementary to the taxonomy of entities and properties underlying the SIMPLE General Ontology. Therefore, this theory has been adapted and conceptualised into OntoTag's LUO as well, as a right and proper way of subspecifying the meaning (*i.e.*, the sense) of verbs and circumstances.
- 2.8. *GUME*: this element references an extension of Bateman's (1990) Generalised Upper Model (GUM) for Spanish, called GUME (Bernardos-Galindo, 1997)¹⁹⁵.
- 2.9. *CNEO*: This is the acronym for the already mentioned Cinema Named Entity Ontology. It is described in detail in Subsection 5.4.3.2.

¹⁹⁵ GUME has been conceptualised and added as a secondary ontology into OntoTag's ontologies. This ontology is not presented here for the sake of space.

This re-definition of word sense identifiers allowed (A) associating multiple alternative senses to a given word, according to different (domain) ontologies or semantic theories¹⁹⁶; (B) a suitable ontological representation of the conceptual projections (the senses) of lexical units, by means of the concepts and instances of the ontologies included in OntoTagger¹⁹⁷; and (C) a language-independent (possibly multilingual) approach to the development of the lexicon, since meanings are treated rather separately from their realisations by the mediation of OntoTag’s linguistic ontologies.

Another main contribution of OntoTaggerLex with respect to the recommendations in EAGLES (1999) is the way in which (semantic) relationships are represented. From an abstract point of view, all of them are established between two entries (senses) in the lexicon and can be characterised (1) by the two entries related and (2) by the type of relationship holding between them. If its type is represented as an attribute of the relationship, then all relationships can be represented by an abstract relation in the lexicon, namely, *is_related_to*, and the value of its attribute *REL_TYPE* differentiates between the variety of semantic relationships that can be identified and represented. Since most of them can be subcategorised, a particular field for their subcategorisation (*TYPE*) was also required to complete the definition of each of them when necessary.

The list of semantic relationships incorporated into OntoTaggerLex is shown in Table 129, specified by means of the two attributes aforementioned, *REL_TYPE* and *TYPE*.

Table 129: A list of the different relationships represented in OntoTaggerLex

REL_TYPE	TYPE
Synonymy	[Not applicable]
N-Synonymy	[Not applicable]
Hyponymy	Exclusive
Hyponymy	Conjunctive
Hyponymy	Non-exclusive
Hyperonymy	Exclusive
Hyperonymy	Conjunctive
Hyperonymy	Non-exclusive
Antonymy	Complementary
Antonymy	Gradable
Antonymy	Pseudo-comparative
Antonymy	True-comparative
Antonymy	Antipodal

¹⁹⁶ In the original document (EAGLES, 1999) the *Sense* attribute of a lexicon *Entry* was not multi-value. However, a given *Entry* could be described by more than one ontology (a domain-specific and a domain-independent one, *i.e.*, CNEO and EuroWordNet, for example) or lexical resource. Therefore, this property of entries was conveniently modified.

¹⁹⁷ This aspect is crucial to semantic annotations for the Semantic Web.

REL_TYPE	TYPE
Antonymy	Reversive
Antonymy	Relational
Meronymy	Member
Meronymy	Substance
Meronymy	Part
Meronymy	Portion
Meronymy	Location
Holonymy	Member
Holonymy	Substance
Holonymy	Part
Holonymy	Portion
Holonymy	Location

After the main extensions and adaptations of EAGLES (1999) included in OntoTaggerLex have been briefly discussed, both the Entity-Relationship (ER) model and the physical design diagram of OntoTaggerLex can be easily understood. They are shown, respectively, in Figure 21 and in Figure 22. The description of the entities and relationships included in these figures that have not been described here can be found in EAGLES (1999). Further details about the design and implementation of OntoTaggerLex can be found in Serradilla-Fernández (2004).

5.4.2. THE INSTANCE SEMANTIC ANNOTATION LAYER

This section describes in detail the OntoTagger process that implements the layer of instance semantic annotation works. Therefore, the subsystem responsible for named entity annotation in OntoTagger (the NERCA subsystem) is presented in this subsection. The architecture of this subsystem, which recognises, (sub)classifies and annotates named entities is included in Figure 23 (see page 268).

As can be observed in Figure 23, the NERCA subsystem shares the inputs of the whole SAMM, shown in Figure 19. Still, there remains one input in Figure 23 that is not shared with the rest of processes of the SAMM, namely the Paraphrase Linguistic Matching (sub)module. This module was devised as a (linguistic) paraphrase rule manager useful for: (a) increasing the number of named entities recognised by the system; and (b) improving their (sub)classification process. The need for such a (sub)module was identified when developing this first prototype on OntoTag's architecture, but it has not been implemented yet. This is the reason why it appears into a dashed rectangle in the aforementioned figure.

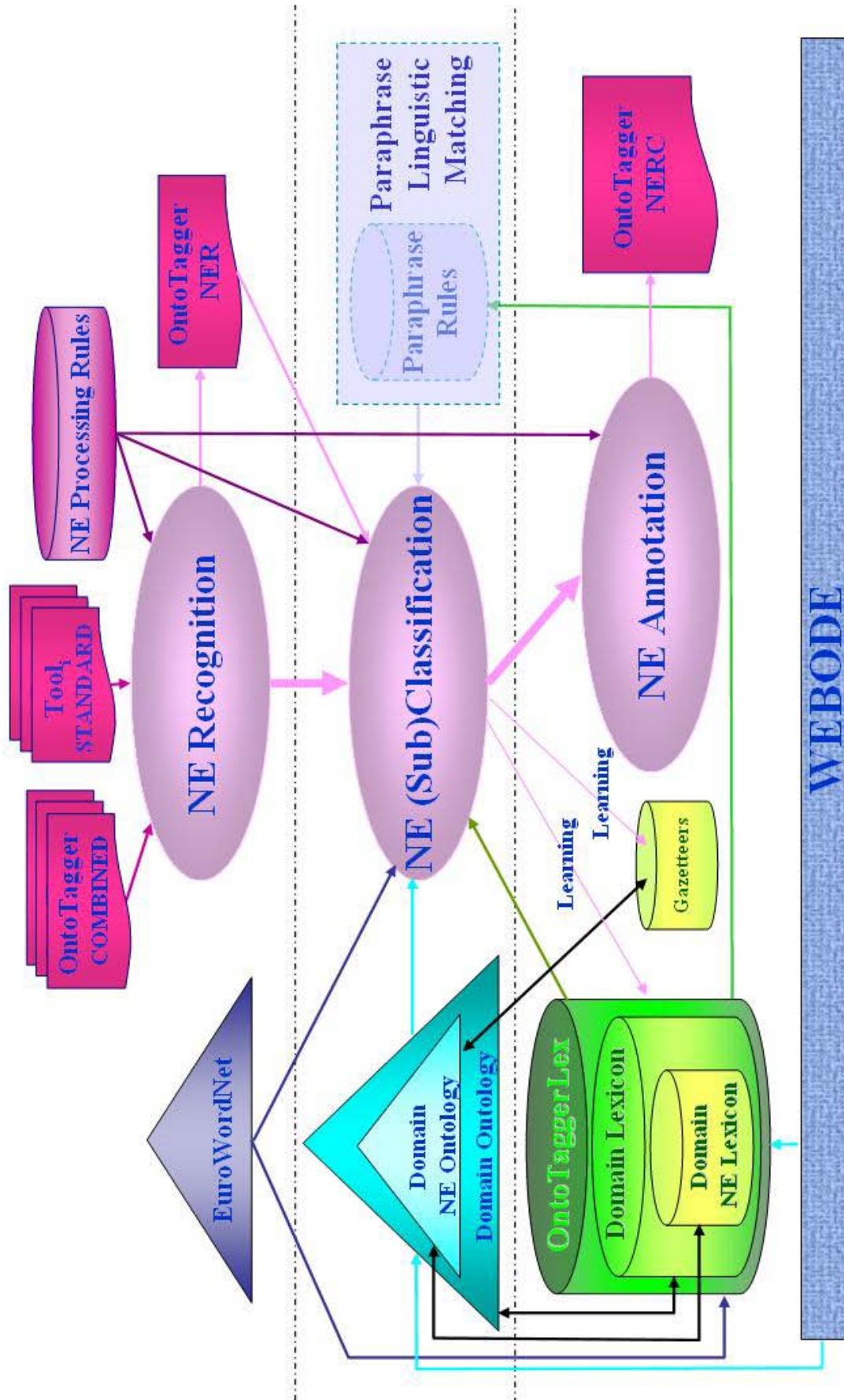


Figure 23: Architecture of the NERCA subsystem

The NERCA subsystem produces two different outputs: (i) an intermediate file (the “OntoTagger NER” file in Figure 23), where the named entities identified by the process of named entity recognition are conveniently marked and which acts as the main input for the process of named entity (sub)classification; and (ii) a new document (the “OntoTagger NERC” document in Figure 23), which contains the annotations of the Instance Semantic Annotation Layer, according to the MUC and ACE tagsets (further specified by means of the CNEO). This latter document is the one to be (intra-level) merged with the rest of the `Semantic Layer` annotation documents afterwards.

The NERCA subsystem consists of three main processes, namely the Named Entity Recognition (NER) process, the Named Entity (Sub)Classification (NEC) process and the Named Entity Annotation (NEA) process:

- The **NER process** is in charge of the identification of the named entities included in the input document, by applying the linguistic heuristics for named entity recognition described in Subsection 5.4.2.1.2. This process takes a `Token` at a time from the input document and first searches for it amongst the instances in the domain ontologies integrated in the system. If the `Token` does not match any of them, then it is looked up in the gazetteers incorporated into the SAMM. Only when none of these two searches ends successfully (*i.e.*, the `Token` is found neither in the sets of instances of the domain ontologies nor in the set of gazetteers) the heuristics for named entity recognition explained in Subsection 5.4.2.1.2 come into play. That is, at this point, these heuristics try to match the `Token` against the set of linguistic patterns that allow identifying Spanish named entities in text. When the current `Token` is recognised as a named entity by any of these three scrutinies, it is tagged as such in the “OntoTagger NER” file and kept in an internal structure to be further processed (in the NEC and in the NEA processes).
- The **NEC process** classifies a given named entity according to the MUC and ACE tagsets and, after that, further subclassifies it according to the domain ontologies incorporated in the system. This is accomplished by means of the tailored heuristics explained in Subsection (5.4.2.1.2), using a similar algorithm as with the one used in the NER process. This might be a bit redundant, but it helps attaining modularity.
- The **NEA process** simply associates a standardised tag to the current named entity, according to the information included in `OntoTag`’s ontologies, and adds this tag to the “OntoTagger NERC” output document associated to the input document.

5.4.2.1 NAMED ENTITY-RELATED LINGUISTIC HEURISTICS

The OEG linguist team¹⁹⁸ identified some linguistic knowledge that can help recognise and subclassify named entities in Spanish. This linguistic knowledge was derived from the study of the already mentioned ODECorpus-Cinema, and has been introduced in the NERCA subsystem via heuristics, formalised as a sort of production rules. These heuristics are supported by the annotations produced by the tools integrated in OntoTagger. Before presenting them, the notation used for the formalisation of these rules will be explained in detail in Section 5.4.2.1.1; the sets of rules for named entity recognition and subclassification are presented afterwards, in sections 5.4.2.1.2.1 and 5.4.2.1.2.2 respectively.

5.4.2.1.1 Description of the Notation

The basic notation used for the specification of the named entity-related linguistic heuristics is very similar to the one used as with Lemma combination (Section 5.3.2.2). However, this basic notation has to be extended with many more functions and procedures in this case. Nevertheless, as with Lemma combination, some of the heuristics for named entity recognition and (sub)classification were determined taking into account the particular tools incorporated in the system and the theoretical grounds underlying their construction. Therefore, the notation concerning the three different tools incorporated into this OntoTag's architecture configuration is the following:

- **NDLPT**: Non-Disambiguating Lemma and POS Tagger (a morphological analyser: **DataLexica**)
- **PLPMT**: Pure Lemma, POS and Morphological Tagger (**LACELL's POS tagger**)
- **LPMT**: Lemma, POS and Morphological Tagger and Parser (**FDG Parser**)
- **SET TOOLS**: Designates the set of external tools integrated into the configuration, *i.e.*, $SET_{TOOLS} = \{NDLPT, PLPMT, LPMT\}$.

Once these primary elements in the notation have been introduced, some others can be defined from them as well:

- **(Meta)Variables.**
 - id_{offset} . This metavariable stores the identifier of the `Token` in the position *offset*, where an $offset = 0$ designates the current token, an $offset = -1$ designates the previous one, an $offset = 1$ designates the following one, etc.

¹⁹⁸ The OEG (Ontological Engineering Group – <http://www.oeg-upm.net>) is a research group from the UPM (Universidad Politécnica de Madrid) within which this thesis has been developed.

- CAT_{NDLPT} , CAT_{PLPMT} , CAT_{LPMTP} . These variables refer to the morphosyntactic category tag assigned by each of these tools to the current `Token`. Their value can be a set or, furthermore, a multiset, taking into account possible ambiguities.
- CAT_t . This metavariable designates one of the three variables described previously, depending on the value of t , where $t \in SET_{TOOLS}$.
- CAT_{COMB} designates the combined morphosyntactic category tag for a `Token`.
- $lemma_{COMB}$. This variable is applied to refer to the combined `Lemma` tag of a `Token`.
- $lemma_t$. This metavariable notates the (set of) `Lemma` tag(s) assigned to a `Token` by the tool t , where $t \in SET_{TOOLS}$.
- *counter*. This `INTEGER` variable is used to construct the `URI` associated to each named entity in the “OntoTagger NERC” output document. This *counter* is set to 0 each time a new document is processed, and it is incremented each time a new named entity is recognised, so that each one has a different identifier.
- []. Anything included between two brackets in the pattern that constitutes the left-hand side of a heuristic rule is an optional element within that pattern (it can appear or not).
- *LIFO*. This is a variable that stores temporarily the tokens that make up a multiword named entity before it is completely processed. It is implemented as a stack or *LIFO* (last-in, first out) computational structure (from where its name in the notation comes).
- *NElist*. This variable stores a list of named entities.
- **Functions.** The different functions required for the specification of the heuristic rules presented in this section, together with their description, are shown in Table 130.

Table 130: Functions used in the specification of the named entity recognition and subclassification heuristics.

FUNCTION HEADER	DESCRIPTION
$HasCat(CAT_i, cat, offset): BOOL$	This function returns <i>TRUE</i> if the category <i>cat</i> is a member of the set CAT_i and returns <i>FALSE</i> otherwise. The input parameter <i>offset</i> is an <code>INTEGER</code> value that identifies the <code>Token</code> being processed (<i>offset</i> = 0 designates the current <code>Token</code> , <i>offset</i> = -1 designates the previous <code>Token</code> , etc.)
$NotEmpty(LIFO): BOOL$	<i>NotEmpty</i> returns <i>TRUE</i> if the <i>LIFO</i> structure includes at least one element, and <i>FALSE</i> in other case (<i>i.e.</i> , when the <i>LIFO</i> is empty)
$isNamedEntity(offset): BOOL$	This function returns <i>TRUE</i> if the <code>Token</code> referenced by <i>offset</i> has already been marked as a named entity. It returns <i>FALSE</i> in other case.

FUNCTION HEADER	DESCRIPTION
<i>IsValue (attribute, value, offset): BOOL</i>	<i>IsValue</i> returns <i>TRUE</i> if the <i>Token</i> pointed out by <i>offset</i> has been tagged by the <i>LPMTTP</i> tool with a value for the input <i>attribute</i> that matches the <i>value</i> input parameter. It returns <i>FALSE</i> in other case.
<i>Is_Lemma(lemma, string, offset): BOOL</i>	This function returns <i>TRUE</i> if the <i>lemma</i> of the <i>Token</i> pointed out by <i>offset</i> is equal to the input <i>string</i> , and returns <i>FALSE</i> otherwise.
<i>Is_Month(lemma, offset): BOOL</i>	<i>Is_Month</i> returns <i>TRUE</i> if the <i>lemma</i> of the <i>Token</i> pointed out by <i>offset</i> equals the name of a month (in Spanish) and <i>FALSE</i> in other case.
<i>isNumber (offset): BOOL</i>	This function returns <i>TRUE</i> if the <i>Token</i> pointed out by <i>offset</i> denotes a number. It returns <i>FALSE</i> in other case.
<i>isRoman (lemma, offset):BOOL</i>	This other function returns <i>TRUE</i> if the <i>lemma</i> of the <i>Token</i> pointed out by <i>offset</i> denotes a Roman number, and <i>FALSE</i> in other case.
<i>Category_Match(CAT₁,CAT₂): BOOL</i>	<i>Category_Match</i> returns <i>TRUE</i> when the morphosyntactic category tags <i>CAT₁</i> and <i>CAT₂</i> share their major word category value, that is, their <i>POS</i> (N, V, AJ, PD, AT, AV, AP, C, NU, I, U, R, or PU – see Section 5.3.2.1.2 for details). It returns <i>FALSE</i> in other case.
<i>MatchesRule(identifier, offset): BOOL</i>	<i>MatchesRule</i> returns <i>TRUE</i> if the <i>Token</i> in the position pointed out by <i>offset</i> has been recognised as a named entity by the rule whose <i>identifier</i> constitutes the first parameter of the function, and <i>FALSE</i> otherwise.
<i>AlreadyClassified (offset): BOOL</i>	This function returns <i>TRUE</i> if the named entity denoted by <i>offset</i> has already been classified previously by any rule. It returns <i>FALSE</i> in other case.
<i>LeadedByCapital (LIFO): BOOL</i>	<i>LeadedByCapital</i> returns <i>TRUE</i> if the <i>Word Form</i> of the <i>Token</i> on top of the <i>LIFO</i> starts with an upper-case letter, and returns <i>FALSE</i> otherwise.
<i>isCapital (string):BOOL</i>	This function returns <i>TRUE</i> if all the characters in the input <i>string</i> are upper-case letters, and <i>FALSE</i> in other case.

FUNCTION HEADER	DESCRIPTION
<i>isLength (string, length): BOOL</i>	This other function returns <i>TRUE</i> when the number of characters in the input <i>string</i> equals <i>length</i> ; it returns <i>FALSE</i> in other case.
<i>isType (type, offset):BOOL</i>	<i>isType</i> returns <i>TRUE</i> if the input <i>type</i> equals the type of named entity of the <code>Token</code> denoted by <i>offset</i> . It returns <i>FALSE</i> otherwise.
<i>IsInSynset (string, lemma, offset): BOOL</i>	This function returns <i>TRUE</i> if the <i>lemma</i> of the <code>Token</code> pointed out by <i>offset</i> is in the same EuroWordNet synset as the input <i>string</i> , and <i>FALSE</i> in other case.

- **Procedures.** The different procedures required for the specification of the heuristic rules presented in this section, together with their description, are shown in Table 131.

Table 131: Procedures used in the specification of the named entity recognition and subclassification heuristics.

PROCEDURE HEADER	DESCRIPTION
<i>Mark_NE (counter)</i>	Within this procedure, the current <code>Token</code> is assigned (i) a label that marks it as a named entity; and (ii) a new named entity identifier generated from <i>counter</i> .
<i>Increase (counter)</i>	This procedure increases in a unit the value of <i>counter</i> (which is initially set to 0)
<i>Push(LIFO, id_token)</i>	This procedure leaves the <code>Token</code> designated by <i>id_token</i> on top of the <i>LIFO</i> .
<i>Empty(LIFO)</i>	<i>Empty</i> resets the <i>LIFO</i> structure (<i>i.e.</i> , no element is left in the <i>LIFO</i> afterwards).
<i>GroupNE (counter, begin, end)</i>	This procedure is called upon a sequence of tokens starting at the offset designated by <i>begin</i> and finishing at the offset designated by <i>end</i> . Within this procedure, this sequence of tokens is grouped into a multiword named entity and assigned a new named entity identifier, generated from <i>counter</i> .
<i>PutTypeNE (type, begin, end)</i>	<i>PutTypeNE</i> assigns a more fine-grained named entity <i>type</i> to the sequence of tokens starting at the offset <i>begin</i> and finishing at the offset <i>end</i> , hence subclassifying it.
<i>SeparateNE (actors, NElist)</i>	This procedure takes as input a list of <i>actors</i> ' names, separated by commas, and returns its corresponding list of named entities <i>NElist</i> .

The rules (or, equivalently, the linguistic heuristics) used in OntoTagger for named entity recognition and (sub)classification are presented in the next subsections, after the elements ((meta)variables, functions and procedures) used for their specification have been conveniently presented. These rules have been grouped according to the linguistic criteria underlying them. Each of them is accompanied by a brief explanation of its meaning (or purpose) and a suitable example of application, extracted from ODECorpus-Cinema. The rules for named entity recognition are presented in Subsection 5.4.2.1.2. Then, the particular rules for named entity subclassification will be presented as well, in Subsection 5.4.2.1.3.

5.4.2.1.2 Rules for Named Entity Recognition

The application of the rules for named entity recognition has been split into two internal stages, for the sake of modularity, namely (i) the identification of simple tokens that (might) constitute by themselves a named entity, and (ii) the aggregation of those simple tokens identified in the previous stage that are in fact part of a multiword named entity. These two stages are explained next in a dedicated Subsection (5.4.2.1.2.1 and 5.4.2.1.2.2, respectively).

5.4.2.1.2.1 RULES FOR SIMPLE NAMED ENTITY IDENTIFICATION

The rules for recognising those simple tokens that (i) constitute a named entity by themselves or (ii) are sound candidates for being part of a multiword named entity are presented in the current subsection. They have been implemented by means of a set of five rules, grouped on the basis of five linguistic criteria (or heuristics). An example of application (extracted from ODECorpus-Cinema) accompanies the specification of set of rules implementing each criterion for the sake of clarity.

- *Criterion NEID-1*: Strings between double or single quotes¹⁹⁹ in the corpus are named entities.

Example: ‘Cyrano de Bergerac’.

- RULE #1: HasCat (CAT_{COMB}, PU11, 0) → Push (LIFO, id₀)
- RULE #2: HasCat (CAT_{COMB}, PU12, 0) → Empty (LIFO) ∧ Increase (counter)
- RULE #3: HasCat (CAT_{COMB}, PU09, 0) → Push (LIFO, id₀)
- RULE #4: HasCat (CAT_{COMB}, PU10, 0) → Empty (LIFO) ∧ Increase (counter)
- RULE #5: LededByCapital (LIFO) ∧ NotEmpty (LIFO) → Mark_NE (counter)

¹⁹⁹ In OntoTagger, double (open and close) quotes are assigned, according to the LUO, the POS combined tag (CAT_{COMB}) PU11 and PU12, respectively. Analogously, single (open and close) quotes are assigned the POS combined tag (CAT_{COMB}) PU09 and PU10, respectively.

- *Criterion NEID-2*: Those tokens whose NDLPT (DataLexica) category tag is RU (Unclassified Residual) and whose LPMTSP (FDG) category tag is NP (Proper Noun) are named entities.

Example: ...en el convulso París de 1830

- RULE #6: $\text{HasCat}(\text{CAT}_{\text{NDLPT}}, \text{RU}, 0) \wedge \text{HasCat}(\text{CAT}_{\text{LPMTSP}}, \text{NP}, 0)$
 $\rightarrow \text{Mark_NE}(\text{counter}) \wedge \text{Increase}(\text{counter})$

- *Criterion NEID-3*: Those tokens whose combined morphosyntactic category tag is a kind of Numeral (that is, they have been assigned a morphosyntactic category tag starting with NU) are named entities.

Example: [...] en el convulso París de 1830

- RULE #7: $\text{Category_Match}(\text{CAT}_{\text{COMB}}, \text{"NU"})$
 $\rightarrow \text{Increase}(\text{counter}) \wedge \text{Mark_NE}(\text{counter})$

- *Criterion NEID-4*: Those tokens whose combined category tag is NP (Proper Noun) are named entities.

Example: Como protagonista, Gerard Depardieu, completamente volcado [...]

- RULE #8: $\text{Category_Match}(\text{CAT}_{\text{COMB}}, \text{"NP"})$
 $\rightarrow \text{Increase}(\text{counter}) \wedge \text{Mark_NE}(\text{counter})$

- *Criterion NEID-5*: Those tokens whose combined Lemma begins with a capital letter and which are not preceded by a Period (‘.’) are named entities.

Example: [...] acaparó premios en Sitges 2001

- RULE #9: $\text{isCapital}(\text{lemma}_{\text{COMB}}) \wedge \neg \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"."}, -1)$
 $\rightarrow \text{Increase}(\text{counter}) \wedge \text{Mark_NE}(\text{counter})$

5.4.2.1.2.2 RULES FOR MULTIWORD NAMED ENTITY AGGREGATION

The rules for aggregating several simple tokens into a multiword named entity are presented in this subsection. They have been implemented by means of a set of five rules, each one corresponding to a particular linguistic criterion (*i.e.*, heuristic). For some of the criteria, an example of application (extracted from ODECORPUS-Cinema, when such an example can be found in this corpus) is presented together with the specification of the rule implementing it.

In a number of these criteria, two particular attributes from the LAO (Surface Tag and Phrase Function) and some of their corresponding values from the LVO are used. The meaning of these attributes and values is explained in Table 132.

Table 132: The meaning and main values of the attributes Surface Tag and Phrase Function.

ATTRIBUTE	VALUE	MEANING
Surface Tag	A	The Token having this value for this attribute plays the role of an Adjective in the context where it appears, even though it might not be an Adjective itself.
	N	The Token having this value for this attribute plays the role of a Noun in the context where it appears, even though it might not be a Noun itself.
Phrase Function	R	The Token having this value for this attribute functions as a Premodifier in the context (the Phrase) where it appears.
	H	The Token having this value for this attribute functions as the Head in the context (the Phrase) where it appears.

- *Criterion NEAG-1:* Let NE_1 and NE_2 be two consecutive named entities. If the value assigned to the Phrase Function attribute by the LPMTSP (FDG Parser) is R (Premodifier) for NE_1 and H (Head) for NE_2 , then both named entities can be grouped together into a unique named entity²⁰⁰.

Example: Gerard Depardieu

- RULE #10: $IsNamedEntity(-1) \wedge IsNamedEntity(0) \wedge$
 $IsValue(phrase_function, R, -1) \wedge IsValue(phrase_function, H, 0)$
 $\rightarrow GroupNE(counter, -1, 0)$

- *Criterion NEAG-2:* Let NE_1 , NE_2 and NE_3 be three consecutive named entities. If the value assigned to the Phrase Function attribute by the LPMTSP (FDG Parser) is R (Premodifier) for NE_1 , R or H (Head) for NE_2 and H for NE_3 , then the three named entities can be grouped together into a unique named entity²⁰¹.

Example: Jean Christophe Comar

²⁰⁰ This rule is applied to several instances of the corpus, in order to join into a single multiword named entity a person's name and surname.

²⁰¹ The first one is usually the first name, the second one a second name, and the third one the surname of a person. Alternatively, the second and the third one can be a person's composite surname.

- RULE #11: $\text{IsNamedEntity}(-2) \wedge \text{IsNamedEntity}(-1) \wedge \text{IsNamedEntity}(0) \wedge$
 $\text{IsValue}(\text{phrase_function}, \text{R}, -2) \wedge \text{IsValue}(\text{phrase_function}, \text{H}, 0) \wedge$
 $(\text{IsValue}(\text{phrase_function}, \text{R}, -1) \parallel \text{IsValue}(\text{phrase_function}, \text{H}, -1))$
 $\rightarrow \text{GroupNE}(\text{counter}, -2, 0)$
- *Criterion NEAG-3*: Let NE_1 and NE_2 be two consecutive named entities. If the value assigned to the `Phrase Function` attribute by the LPMTSP (FDG Parser) for both of them is `R` (`Premodifier`), then NE_1 and NE_2 can be grouped together into a unique named entity²⁰².
- RULE #12: $\text{IsNamedEntity}(-1) \wedge \text{IsNamedEntity}(0) \wedge$
 $\text{IsValue}(\text{phrase_function}, \text{R}, -1) \wedge \text{IsValue}(\text{phrase_function}, \text{R}, 0)$
 $\rightarrow \text{GroupNE}(\text{counter}, -1, 0)$
- *Criterion NEAG-4*: Let NE_1 and NE_2 be two consecutive named entities. If the value assigned to the `Phrase Function` attribute by the LPMTSP (FDG Parser) for both of them is `H` (`Head`), then NE_1 and NE_2 can be grouped together into a unique named entity²⁰³.
- RULE#13: $\text{IsNamedEntity}(-1) \wedge \text{IsNamedEntity}(0) \wedge$
 $\text{IsValue}(\text{phrase_function}, \text{H}, -1) \wedge \text{IsValue}(\text{phrase_function}, \text{H}, 0)$
 $\rightarrow \text{GroupNE}(\text{counter}, -1, 0)$
- *Criterion NEAG-5*: If a sequence of tokens matches the pattern

$$\boxed{NE_1 + \text{'de'} + NE_2 + \text{'de'} + NE_3}$$

where both NE_1 and NE_3 are cardinal numbers and NE_2 is the name of a month (in Spanish), they can be grouped together into a unique named entity (which, furthermore, can be subclassified as a `Date`).

Example: 25 de enero de 2002.

- RULE #14: $\text{HasCat}(\text{CAT}_{\text{COMB}}, \text{NUC}, -4) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{'de'}, -3) \wedge$
 $\text{Is_Month}(\text{lemma}_{\text{COMB}}, -2) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{'de'}, -1) \wedge$
 $\text{HasCat}(\text{CAT}_{\text{COMB}}, \text{NUC}, 0)$
 $\rightarrow \text{GroupNE}(\text{counter}, -4, 0) \wedge \text{PutTypeNE}(\text{Date}, -4, 0)$

After the different named entities in the input text have been recognised by means of the rules presented in this subsection, they must be (sub)classified according to `OntoTag`'s ontologies or other

²⁰² No evidence supporting this rule was found in the referenced corpus. It was added for completeness sake.

²⁰³ Idem.

domain ontology (the CNEO, in this case). This is achieved in OntoTagger by applying the set of rules for named entity (sub)classification described in the next subsection.

5.4.2.1.3 Rules for Named Entity (Sub)Classification

Hence, the rules for (sub)classifying the named entities already detected in the NERCA subsystem by the set of rules described in the previous section are presented in this subsection. They implement the most relevant set of criteria for named entity processing and annotation included in OntoTagger. An example of application (extracted from ODECorpus-Cinema) accompanies the description of the set of rules implementing those criteria most frequently applied.

- *Criterion NESC-1*: When a sequence of tokens matches any of the following two patterns:

NE + ‘minutos’
NE + ‘min’

then NE can be subclassified as a `Film duration` named entity.

Example: Duración: 120 min.

- RULE #15.1: $\text{MatchesRule}(\text{“rule7”}, 0) \wedge \text{IsNamedEntity}(0) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“minutos”}, 1) \rightarrow \text{PutTypeNE}(\text{Film duration}, 0, 0)$
- RULE #15.2: $\text{MatchesRule}(\text{“rule7”}, 0) \wedge \text{IsNamedEntity}(0) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“min”}, 1) \rightarrow \text{PutTypeNE}(\text{Film duration}, 0, 0)$

- *Criterion NESC-2*: Those named entities appearing in the corpus between single or double quotes can be subclassified as `Film` named entities.

Example: ‘Matrimonio de conveniencia’

- RULE #16: $\text{MatchesRule}(\text{“rule5”}, 0) \wedge \text{IsNamedEntity}(0) \rightarrow \text{PutTypeNE}(\text{Film}, 0, 0)$

- *Criterion NESC-3*: This criterion comprises a set of very particular (and corpus-dependent) heuristics for named entity subclassification based on some film credit lists that appear in the corpus.

Example 1: Título de la película: El Perdón

- RULE #17.1: $(\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"Título"}, -5) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"de"}, -4) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"la"}, -3) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"película"}, -2) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{":"}, -1) \rightarrow \text{PutTypeNE}(\text{Translated Title Entity}, 0, \text{EOL}))$

Example2: Título original / T. O: The Claim

- RULE #17.2.1: $(\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"Título"}, -3) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"original"}, -2) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{":"}, -1) \rightarrow \text{PutTypeNE}(\text{Original Title Entity}, 0, \text{EOL}))$
- RULE #17.2.2: $(\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"T.O."}, -2) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{":"}, -1) \rightarrow \text{PutTypeNE}(\text{Original Title Entity}, 0, \text{EOL}))$

Example3: Director: Michael Winterbottom

- RULE #17.3: $(\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"Director"}, -2) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{":"}, -1) \rightarrow \text{PutTypeNE}(\text{Director}, 0, \text{EOL}))$

Example 4: Intérpretes: Peter Mullan, Nastassia Kinski y Sarah Polley.

- RULE #17.4: $(\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"intérprete"}, -2) \wedge (\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{":"}, -1) \rightarrow \text{PutTypeNE}(\text{Actor}, 0, \text{EOL}) \wedge \text{SeparateNE}(\text{actors}, \text{NElist}))$
- *Criterion NES-4: When a sequence of tokens matches the pattern*

$[\text{'en' | 'de'}] + \text{NE}$

where NE is a numerical named entity denoting a year, this allows for the (sub)classification of NE as a Date.

- RULE #18.1: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"de"}, -1) \wedge \text{isLength}(\text{lemma}_{\text{COMB}}, 4) \wedge \text{HasCat}(\text{CAT}_{\text{COMB}}, \text{NUC}, 0) \wedge \text{isNumber}(0) \rightarrow \text{PutTypeNE}(\text{Date}, 0, 0)$
- RULE #18.2: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"en"}, -1) \wedge \text{isLength}(\text{lemma}_{\text{COMB}}, 4) \wedge \text{HasCat}(\text{CAT}_{\text{COMB}}, \text{NUC}, 0) \wedge \text{isNumber}(0) \rightarrow \text{PutTypeNE}(\text{Date}, 0, 0)$

- *Criterion NESC-5*: A sequence of tokens that matches the pattern

‘nacer’ + ‘en’ + NE

where NE is a non-numerical named entity, allows for the classification of NE as a Geographical Location.

- RULE #19: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“nacer”}, -2) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“en”}, -1) \wedge \text{IsNamedEntity}(0) \wedge \neg \text{HasCat}(\text{CAT}_{\text{COMB}}, \text{NUC}, 0) \rightarrow \text{PutTypeNE}(\text{Geographical Location}, 0, 0)$

- *Criterion NESC-6*: When a sequence of tokens matches the following pattern:

[‘localidad’ | ‘ciudad’ | ‘región’ | ‘villa’ | ‘comarca’ | ...] + [] + ‘de’ + NE

the NE can be classified as a Geographical Location.

- RULE #20.1: $(\text{IsInSynset}(\text{“localidad”}, \text{lemma}_{\text{COMB}}, -3) \vee \text{IsInSynset}(\text{“ciudad”}, \text{lemma}_{\text{COMB}}, -3) \vee \text{IsInSynset}(\text{“región”}, \text{lemma}_{\text{COMB}}, -3) \vee \text{IsInSynset}(\text{“villa”}, \text{lemma}_{\text{COMB}}, -3) \vee \text{IsInSynset}(\text{“comarca”}, \text{lemma}_{\text{COMB}}, -3)) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“de”}, -1) \wedge \text{IsNamedEntity}(0) \rightarrow \text{PutTypeNE}(\text{Geographical Location}, 0, 0)$
- RULE #20.2: $(\text{IsInSynset}(\text{“localidad”}, \text{lemma}_{\text{COMB}}, -2) \vee \text{IsInSynset}(\text{“ciudad”}, \text{lemma}_{\text{COMB}}, -2) \vee \text{IsInSynset}(\text{“región”}, \text{lemma}_{\text{COMB}}, -2) \vee \text{IsInSynset}(\text{“villa”}, \text{lemma}_{\text{COMB}}, -2) \vee \text{IsInSynset}(\text{“comarca”}, \text{lemma}_{\text{COMB}}, -2)) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“de”}, -1) \wedge \text{IsNamedEntity}(0) \rightarrow \text{PutTypeNE}(\text{Geographical Location}, 0, 0)$

- *Criterion NESC-7*: A sequence of tokens that matches the pattern

‘los’ + ‘años’ + NE

where NE is a numerical named entity, allows for the classification of NE as a Date.

- RULE #21: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“año”}, -1) \wedge \text{IsNamedEntity}(0) \wedge \text{HasCat}(\text{CAT}_{\text{COMB}}, \text{NUC}, 0) \rightarrow \text{PutTypeNE}(\text{Date}, 0, 0)$

- *Criterion NESC-8*: When a sequence of tokens matches the following pattern:

$$\text{'entre' + NE}_1 + \text{'y' + NE}_2 + [\cdot]$$

where both NE_1 and NE_2 are numerical named entities, and $[\cdot]$ is any type of Token except for a Common Noun (NC), both NE_1 and NE_2 can be classified as a Date.

- RULE #22: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{'entre'}, -4) \wedge \text{IsNamedEntity}(-3)$
 $\wedge \text{isType}(\text{NUC}, -3) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{'y'}, -2)$
 $\wedge \text{IsNamedEntity}(-1) \wedge \text{HasCat}(\text{CAT}_{\text{COMB}}, \text{NUC}, -1)$
 $\wedge \neg \text{HasCat}(\text{CAT}_{\text{COMB}}, \text{NC}, 0)$
 $\rightarrow \text{PutTypeNE}(\text{Date}, -3, -3) \wedge \text{PutTypeNE}(\text{Date}, -1, -1)$

- *Criterion NESC-9*: A sequence of tokens that matches the pattern

$$\text{NE}_1 + \text{'(' + NE}_2$$

allows subclassifying NE_2 as an Actor.

- RULE #23: $\text{IsNamedEntity}(-2) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{'('}, -1) \wedge \text{IsNamedEntity}(0)$
 $\rightarrow \text{PutTypeNE}(\text{Actor}, 0, 0)$

- *Criterion NESC-10*: When a sequence of tokens matches the following pattern:

$$\text{NE}_1 + \text{'(' + ' ' + NE}_2$$

where NE_2 is a Film named entity, then NE_1 can be subclassified as an Actor.

- RULE #24: $\text{IsNamedEntity}(-3) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{'('}, -2) \wedge$
 $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{' '}, -1) \wedge \text{IsNamedEntity}(0) \wedge \text{isType}(\text{Film}, 0)$
 $\rightarrow \text{PutTypeNE}(\text{Actor}, -3, -3)$

- *Criterion NESC-11*: A sequence of tokens that matches the pattern

$$\text{'Festival' + 'de' + NE}$$

allows for the aggregation and the subclassification of the three tokens in question into a unique named entity of type Cinema Festival.

- RULE #25: $\text{IsInSynset}(\text{'festival'}, \text{lemma}_{\text{COMB}}, -2) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{'de'}, -1)$
 $\wedge \text{IsNamedEntity}(0)$
 $\rightarrow \text{PutTypeNE}(\text{Cinema Festival}, -2, 0)$

- *Criterion NESC-12*: When a sequence of tokens matches the following pattern:

‘Premio’ + ‘en’ + NE ₁ + NE ₂

where NE₂ is a numerical named entity, this allows aggregating and subclassifying NE₁ and NE₂ into a unique named entity of type `Cinema Festival`.

- **RULE #26**: $\text{IsInSynset}(\text{“premio”}, \text{lemma}_{\text{COMB}}, -2) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“en”}, -1)$
 $\wedge \text{IsNamedEntity}(0) \wedge \text{IsNamedEntity}(1) \wedge \text{HasCat}(\text{CAT}_{\text{COMB}}, \text{NUC}, 0)$
 $\rightarrow \text{PutTypeNE}(\text{Cinema Festival}, 0, 1)$

- *Criterion NESC-13*: A sequence of tokens that matches the pattern

‘novela’ + [·] + NE

where [·] is any type of (non-empty) `Token`, allows subclassifying NE as a `Writer`.

- **RULE #27**: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“novela”}, -3) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“de”}, -1)$
 $\wedge \text{IsNamedEntity}(0)$
 $\rightarrow \text{PutTypeNE}(\text{Writer}, 0, 0)$

- *Criterion NESC-14*: When a sequence of tokens matches the following pattern:

‘director’ + [] + ‘como’ + NE

NE can be subclassified as a `Director`.

- **RULE #28.1**: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“director”}, -3)$
 $\wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“como”}, -1) \wedge \text{IsNamedEntity}(0)$
 $\rightarrow \text{PutTypeNE}(\text{Director}, 0, 0)$
- **RULE #28.2**: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“director”}, -2)$
 $\wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{“como”}, -1) \wedge \text{IsNamedEntity}(0)$
 $\rightarrow \text{PutTypeNE}(\text{Director}, 0, 0)$

- *Criterion NESC-15*: A sequence of tokens that matches any of the following patterns:

‘dirigir’ + ‘por’ + NE

‘dirigir’ + ‘en’ + [] + ‘por’ + NE

allows subclassifying NE as a `Director`.

- RULE#29.1: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"dirigir"}, -2) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"por"}, -1) \wedge \text{IsNamedEntity}(0)$
 $\rightarrow \text{PutTypeNE}(\text{Director}, 0, 0)$
- RULE#29.2: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"dirigir"}, -4) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"en"}, -3) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"por"}, -1) \wedge \text{IsNamedEntity}(0)$
 $\rightarrow \text{PutTypeNE}(\text{Director}, 0, 0)$
- *Criterion NESC-16*: When a sequence of tokens matches the pattern

$$\boxed{\text{NE}_1 + [] + \text{' , ' + 'de' + NE}_2}$$

where NE_1 is a `Film` named entity, NE_2 can be subclassified as a `Director`.

- RULE#30.1: $\text{IsNamedEntity}(-3) \wedge \text{isType}(\text{"Film"}, -3) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{" , "}, -2) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"de"}, -1) \wedge \text{IsNamedEntity}(0)$
 $\rightarrow \text{PutTypeNE}(\text{Director}, 0, 0)$
- RULE#30.2: $\text{IsNamedEntity}(-4) \wedge \text{isType}(\text{"Film"}, -4) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{" , "}, -2) \wedge \text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"de"}, -1) \wedge \text{IsNamedEntity}(0)$
 $\rightarrow \text{PutTypeNE}(\text{Director}, 0, 0)$
- *Criterion NESC-17*: A sequence of tokens that matches the pattern

$$\boxed{\text{'siglo' + } \alpha}$$

where α can be interpreted as a Roman number, allows for the aggregation and subclassification of both tokens into a unique named entity of type `Century`.

- RULE#31: $\text{Is_Lemma}(\text{lemma}_{\text{COMB}}, \text{"siglo"}, -1) \wedge \text{isRoman}(\text{lemma}_{\text{COMB}}, 0)$
 $\rightarrow \text{PutTypeNE}(\text{Century}, -1, 0)$
- *Criterion NESC-18*: Those named entities already subclassified in the annotation of the input document or included as instances of a concept in the domain ontology are assigned the same subclassification in ulterior occurrences.
- RULE #32: $\text{AlreadyClassified}(0) \rightarrow \text{PutTypeNE}(\text{type}, \text{begin}, \text{end})$

Hence, the whole set of rules for the recognition, aggregation and subclassification of named entities within `OntoTagger` has been presented thus far. The dependency of this set of rules with respect to the language, the domain and the corpus chosen was analysed after they had been developed. The result of this analysis is shown in Table 133. For each rule, in a dedicated column, it has been indicated in this table whether it is language-dependent, domain-dependent and corpus-dependent or not.

Table 133: A survey of the dependency properties of the named entity processing heuristics

RULE IDENTIFIER	LANGUAGE-DEPENDENT	DOMAIN-DEPENDENT	CORPUS-DEPENDENT	TOOL-DEPENDENT
RULE #1	NO	NO	NO	NO
RULE #2	NO	NO	NO	NO
RULE #3	NO	NO	NO	NO
RULE #4	NO	NO	NO	NO
RULE #5	NO	NO	NO	NO
RULE #6	NO	NO	NO	YES
RULE #7	NO	NO	NO	NO
RULE #8	NO	NO	NO	NO
RULE #9	YES	NO	NO	NO
RULE #10	NO	NO	NO	YES
RULE #11	NO	NO	NO	YES
RULE #12	NO	NO	NO	YES
RULE #13	NO	NO	NO	YES
RULE #14	YES	NO	NO	NO
RULE #15.1	YES	YES	NO	NO
RULE #15.2	YES	YES	NO	NO
RULE #16	YES	YES	NO	NO
RULE #17.1	YES	YES	YES	NO
RULE #17.2.1	YES	YES	YES	NO
RULE #17.2.2	YES	YES	YES	NO
RULE #17.3	YES	YES	YES	NO
RULE #17.4	YES	YES	YES	NO
RULE #18.1	YES	YES	NO	NO
RULE #18.2	YES	YES	NO	NO
RULE #19	YES	NO	NO	NO
RULE #20.1	YES	NO	NO	NO
RULE #20.2	YES	NO	NO	NO
RULE #21	YES	NO	NO	NO
RULE #22	YES	NO	NO	NO
RULE #23	NO	YES	YES	NO
RULE #24	NO	YES	YES	NO
RULE #25	YES	YES	NO	NO
RULE #26	YES	YES	NO	NO
RULE #27	YES	NO	NO	NO
RULE #28.1	YES	NO	NO	NO
RULE #28.2	YES	NO	NO	NO
RULE #29.1	YES	NO	NO	NO
RULE #29.2	YES	NO	NO	NO
RULE #30.1	YES	YES	NO	NO
RULE #30.2	YES	YES	NO	NO
RULE #31	YES	NO	NO	NO
RULE #32	NO	YES	NO	YES

As can be observed in Table 133, there are in total 42 rules implemented within the NERCA subsystem; out of them, (i) 27 are language-dependent, (ii) 17 are domain-dependent, (iii) 7 are corpus-dependent and (iv) 6 are tool-dependent.

This sort of dependency analysis of the rules implemented within the NERCA subsystem for the recognition, aggregation and annotation of named entities concludes the presentation of the whole NERCA subsystem itself and of the Instance Semantic Annotation Layer included in OntoTagger. Another important layer of semantic annotation included in this system, namely the Concept Semantic Annotation Layer, is described in the following section.

5.4.3. THE CONCEPT SEMANTIC ANNOTATION LAYER

As commented in the introduction of Section 5.4, adverbs are the only lexical units sense-tagged by the linguistic annotation tools incorporated into OntoTagger. Therefore, the main goal of the Concept Semantic Annotation Layer of OntoTagger was to extend this sense tagging sub-layer to other lexical units. As shown in Figure 20, this goal was attained by the implementation in the SAMM of two different processes, namely the Domain-Independent Sense Tagging (DIST) process and the Domain-Dependent Sense Tagging (DDST) process. Each of these processes is discussed in more detail in the following two subsections.

5.4.3.1 DOMAIN-INDEPENDENT SENSE TAGGING: *EUROWORDNET* INTEGRATION

As already mentioned, the DDST process attaches a domain independent²⁰⁴ sense tag to a number of the tokens identified in the input document. First, it is considered domain-independent, provided that it is performed in terms of a general purpose lexical ontology, *i.e.*, EuroWordNet²⁰⁵. Second, only a number of the input tokens can be assigned a tag within this process, provided that this linguistic resource includes only content words. Therefore, non-content words cannot be tagged within this process. Lastly, it is said to be non-disambiguated, provided that no Word Sense Disambiguation task has been implemented within this process (though it has been devised its inclusion).

Accordingly, since no disambiguation is applied in this process, its work is very simple. Each word found in the “Syn COMBINED” document, resulting from the combination of the syntactic annotations of all the tools integrated in OntoTagger (see Figure 16), is looked up in EuroWordNet. If it is included in any of its synsets, then the identifier(s) of this/these synset(s) is/are retrieved and

²⁰⁴ Possibly ambiguous.

²⁰⁵ The integration of EuroWordNet within OntoTagger has been implemented using the *EuroWordNet Java API*, release 0.6.

inserted in a list²⁰⁶. Finally, this (possibly unitary) list of synset identifiers is attached to that content word as a sense tag in the intermediate file containing the sense-tagging of the input document being annotated.

5.4.3.2 DOMAIN-DEPENDENT SENSE TAGGING: CNEO INTEGRATION

In addition, the Concept Semantic Annotation Layer implemented in the SAMM includes a process that adds a domain dependent sense tagging to the input document. In this type of sense tagging, each semantic element particular to the domain is assigned a tag that is a reference to its corresponding concept of the Cinema Named Entities Ontology (CNEO).

As has been commented above, the CNEO is a domain ontology created on purpose for the conceptualisation of the domain to which OntoTagger was eventually applied (basically film reviews). This conceptualisation was obtained from a terminological study of the corresponding sub-corpus of ODECorpus-Entertainment, that is, the ODECorpus-Cinema sub-corpus. A graphical representation of the CNEO is shown in Figure 24 (page 287).

The top concept of this ontology (standing at level 0) is `Named Entity`, as the aforementioned figure shows. This top-level concept is imported from the Linguistic Unit Ontology (LUO) of the model OntoTag and establishes the relationship between this domain ontology and the LUO. Hence, from this point of view, the LUO acts as a top-level ontology for (domain) named entity annotation. Below the top concept in the taxonomy hangs the classification of named entities distinguished within the MUC and the ACE tagsets.

This classification occupies the following two levels, and also partly the next level (the third one) of the taxonomy tree. The concepts of this classification in the first level are `Proper Name Entity`, `Number Expression` and `Temporal Expression`, and `Time`, `Date`, amongst others, in the second level. Below this classification, starting also at the third level, the different subclassifications of cinema named entities that have been found in ODECorpus-Cinema were conveniently formalised.

This same ontology is also shown in Figure 25 (page 288), but this time making explicit the attribute that identifies unambiguously each concept in the ontology. Each concept can be referenced by means of the values of this attribute in any intermediate code tagging developed as the one recommended for morphosyntactic annotation in EAGLES (1996a).

²⁰⁶ As stated above, a word can be included in more than one synset if it is ambiguous, either morphosyntactically or semantically.

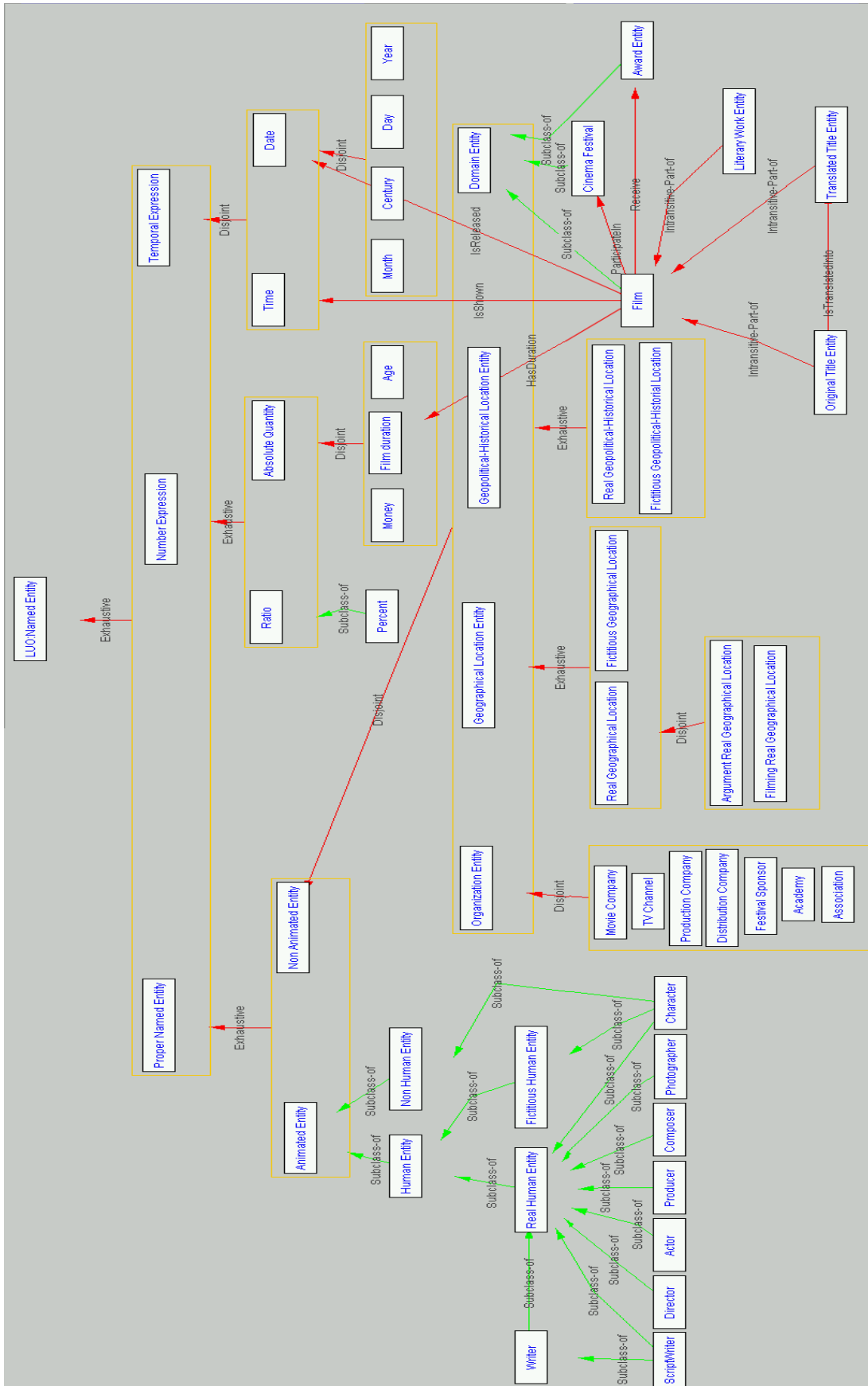


Figure 24: A graphical representation of the Cinema Named Entity Ontology (CNEO)

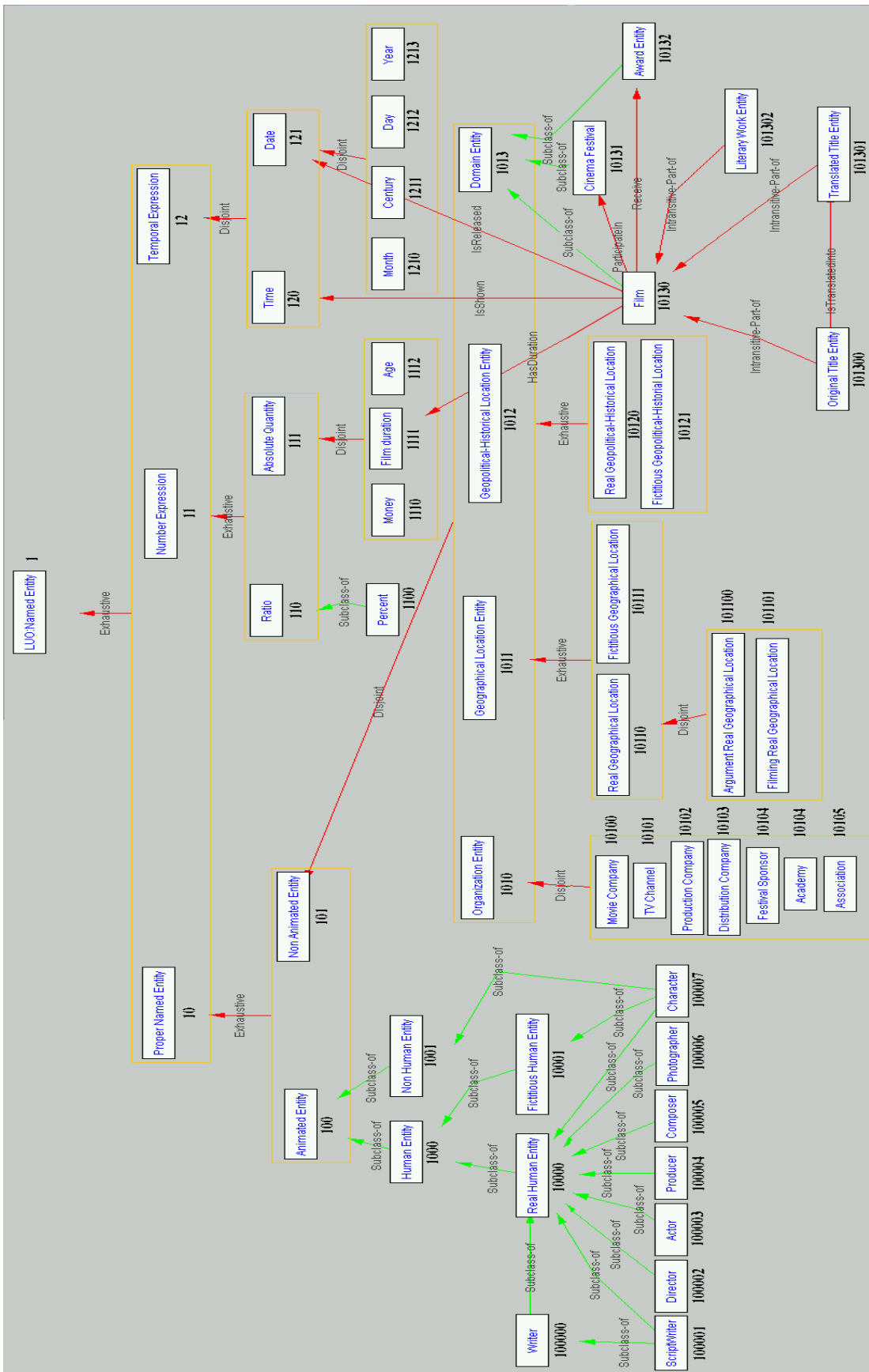


Figure 25: CNEO – the attribute numbering

The way in which a concept is identified by means of this attribute is based on the hierarchical structure of the CNEO. In general, if n is the level at which a concept stands (being 0 the level of the top/root concept, `Named Entity`), then it can be identified by means of $n + 1$ digits. Each concept of a given level ‘inherits’ the identifier of its parent concept, which lies at the previous level of the ontology. Hence, the identifier of each concept consists of its parent concept identifier and an additional digit.

This additional digit is appended to its identifier in order to differentiate it from its siblings. It is assigned to the different children of a concept from left to right, consecutively, starting from 0. For example, following this algorithm, the top concept (`Named Entity`) was assigned a tag containing a single digit (**1**, in this case, breaking exceptionally the general rule). Accordingly, its children, namely `Proper Name Entity`, `Number Expression` and `Temporal Expression`, were assigned **10**, **11** and **12**, respectively; and **100** and **101** were the codes assigned to `Animated Entity` and `Non-Animated Entity`, children of `Proper Named Entity`.

Thus, the CNEO was used within the Concept Semantic Annotation Layer of OntoTagger for sense tagging in the following way. Each CNEO sense tag consists of a pair (i) whose first component is the namespace assigned to the ontology (CNEO, in this case) and (ii) whose second component is the concatenation of the identifier and the name of a CNEO concept²⁰⁷. In other words, the second component references a CNEO concept to which the current Token can be related. This relationship is established as follows. First of all, this sense tag is assigned after the domain-independent sense-tagging process has finished tagging the Token in question. Consequently, for each Token in the input document, the different EuroWordNet synsets with which its Lemma matches are already available. Thus, if any of the synset identifiers matches one of the domain ontology terms, then a pair `<ontology_namespace, concept_identifier#concept_name>` is generated and attached to the Token as a sense-tag in the sense-tagging intermediate file of OntoTagger.

5.4.4. PUTTING SEMANTIC ANNOTATIONS TOGETHER: THE SEMANTIC INTRA-LEVEL MERGER (SILM)

As commented, the Semantic Intra-Level Merger (SILM) is the process responsible for the internal combination of all the layers and sub-layers of semantic annotation incorporated within OntoTagger (presented above). However, as indicated at the beginning of Section 5.4, the NERCA subsystem (that is, the Instance Semantic Annotation Layer) is in charge of guiding the whole process of semantic annotation within OntoTagger. Thus, as the NERCA subsystem is being executed, (i) whenever a new

²⁰⁷ See, for example, the tag associated to ‘Director’ in the Phrase ‘Director: Chris Columbus’ in Table 139, page 306.

Token in the input document is assigned an open-class POS tag (*i.e.*, it is a Noun, a Verb, an Adjective or an Adverb), the Domain-Independent Sense Tagging (DIST) process is called; (ii) whenever a candidate domain concept is identified, the Domain-Dependent Sense Tagging (DDST) process is invoked²⁰⁸; and (iii) whenever a role tag has been attached by Connexor's FDG Parser to a Token, the Circumstantial Role Labelling process adds this tag to the intra-level combined document generated by the SILM.

Hence, in OntoTagger, this process is merely a coordinator of the execution of the three main processes of semantic annotation. In effect, it has to synchronise them on the basis of the Token being currently annotated, so that all of them process the same Token at a time.

5.4.5. THE SEMANTIC LEARNING PROCESS

Finally, the Semantic Learning process included in OntoTagger is presented in this subsection. As mentioned in the introduction of Section 5.4, the main aims of this module are: (i) populating the domain ontology with the instances identified within the process of named entity recognition and classification; (ii) updating the data in the semantic lexicon with the information extracted from EuroWordNet in the process of annotation; and (iii) updating also the information in the gazetteers included in this configuration when a new named entity, not included in them yet, is found in the input text.

Accordingly, this process must work together with the NERCA subsystem, as the rest of processes within the Semantic Annotation Manager Module (SAMM). Whenever a named entity or another piece of semantic information is found, the Semantic Learning (SL) process is called (i) to store this piece of semantic information conveniently, in the domain ontology and/or in the semantic lexicon (OntoTaggerLex, in this case); and (ii) to update the corresponding gazetteer, when necessary.

The description of the final process of semantic integration included in OntoTagger concludes the description of this OntoTag's configuration. The particular annotation schemas developed for integrating in a unique document all the annotations obtained by means of OntoTagger for a given input document are presented in the next section.

²⁰⁸ This is how these two sub-layers of the Concept Semantic Annotation Layer can be automatically combined with the annotations pertaining to the Instance Semantic Annotation Layer.

5.5. ONTOTAGGER'S ANNOTATION SCHEMAS

This section introduces the most salient aspects of the implementation of OntoTag's (abstract) annotation scheme in OntoTagger. This implementation, in fact, consists of three different and alternative implementations or schemas: there is an XML-based implementation, an RDF-based implementation and an OWL-based implementation. Each of these implementations is referred to as an OntoTagger's annotation schema in the present dissertation. For the sake of brevity, only the XML- and the OWL-based implementations are shown here. The RDF(S) implementation underlies the OWL implementation and presenting it as well would be extremely redundant.

An important remark should be made about OntoTagger's annotation schemas. Unfortunately, due to time and human resources restrictions, these schemas might suffer from certain language-specificity (with respect to Spanish, in particular)²⁰⁹. What is certain is that, as will be remarked later on, these schemas suffer from an evident tool-dependency with respect to the different tools incorporated in OntoTagger's architecture configuration (namely Bitext's DataLexica, Connexor's FDG and LACELL's POS tagger). This will be evidenced by some particular attributes described below.






As for the XML- and the OWL-based implementations, they are presented by means of an annotated example (for clarity and expressivity reasons) in subsections 5.5.1 and 5.5.2, respectively.

5.5.1. XML IMPLEMENTATION: AN EXAMPLE

The structure of every OntoTagger-XML-annotated document is presented in this subsection. As mentioned above, the XML Schema according to which these XML-annotated documents are generated (and validated) is shown in APPENDIX A: OntoTagger's Annotation XML Schema. This subsection is based on a particular example of annotation taken from the annotated counterpart of the so-called ODECorpus-Cinema corpus. This example (*i.e.*, the annotation of the **word** 'Director' in the **Phrase** 'Director: Chris Columbus') is shown in Table 134. The information in Table 134 is accompanied by graphical explanations describing the type of the annotations, namely morphosyntactic annotations (which embed the Lemma tags), the remaining syntactic annotations and the semantic annotations.

²⁰⁹ The development of OntoTagger focused on the interoperation of the particular tools available, which annotated Spanish texts. It should be further investigated and evaluated whether the eventual tagset and the design principles adopted in OntoTagger are general enough to be applied as such in other languages, though it should be possible.

Table 134: OntoTagger-XML-based annotation of the Spanish word *director*

<pre><luo:token id="t:10_1_1" DataLexica_id="t:10_1_1" FDG_id="t:8_2_16" UMurcia_id="t:1_2_16"></pre>	
<pre><luo:word id="w:10_1_1_1"></pre>	
<pre><luo:text>Director</luo:text></pre>	
<pre><luo:lemma>director</luo:lemma></pre>	
<pre><llo:syntax luo:category="luo:TS"></pre>	
<pre><lao:surface_tag>lvo:N</lao:surface_tag></pre>	
<pre><lao:phrase_function>lvo:H</lao:phrase_function></pre>	
<pre><lao:morpho-syntactic_function>lvo:N</lao:morpho-syntactic_function></pre>	
<pre><luo:depends_syntactically_on head="w:9_1_2_1">main</luo:depends_syntactically_on></pre>	
<pre></llo:syntax></pre>	
<pre><llo:morpho luo:category="luo:NC"></pre>	
<pre><lao:gender>lvo:M</lao:gender></pre>	
<pre><lao:number>lvo:S</lao:number></pre>	
<pre><lao:case>lvo:N</lao:case></pre>	
<pre><lao:definiteness>0</lao:definiteness></pre>	
<pre></llo:morpho></pre>	
<pre><llo:semantics luo:category="luo:E"></pre>	
<pre><lao:use>lvo:[A P]</lao:use></pre>	
<pre><luo:word_meaning></pre>	
<pre><luo:word_sense luo:resource="CNEO" luo:term_id="100002#Director"/></pre>	
<pre><luo:word_sense luo:resource="EWN" luo:term_id="5921693"/></pre>	
<pre><luo:word_sense luo:resource="EWN" luo:term_id="6013928"/></pre>	
<pre><luo:word_sense luo:resource="EWN" luo:term_id="6048640"/></pre>	
<pre><luo:word_sense luo:resource="EWN" luo:term_id="6279719"/></pre>	
<pre><luo:word_sense luo:resource="EWN" luo:term_id="6282130"/></pre>	
<pre><luo:word_sense luo:resource="EWN" luo:term_id="6369483"/></pre>	
<pre></luo:word_meaning></pre>	
<pre><luo:is_related_to luo:rel_type="luo:synonymy"></pre>	
<pre><luo:unit EWN:term_id="5921693" EWN:lang="ES"></pre>	
<pre><luo:synonymy>gerente</luo:synonymy></pre>	
<pre><luo:synonymy>manager</luo:synonymy></pre>	
<pre></luo:unit></pre>	
<pre><luo:unit EWN:term_id="6013928" EWN:lang="ES"></pre>	
<pre><luo:synonymy>director de orquesta</luo:synonymy></pre>	
<pre></luo:unit></pre>	
<pre>...</pre>	
<pre></luo:is_related_to></pre>	
<pre><luo:is_related_to luo:rel_type="luo:hyperonymy"></pre>	
<pre><luo:unit EWN:term_id="5921424" luo:type="null" EWN:lang="ES"></pre>	
<pre><luo:hyperonymy>administrador</luo:hyperonymy></pre>	
<pre></luo:unit></pre>	
<pre><luo:unit EWN:term_id="6010872" luo:type="null" EWN:lang="ES"></pre>	
<pre><luo:hyperonymy>miembro del comité</luo:hyperonymy></pre>	
<pre></luo:unit></pre>	
<pre></luo:is_related_to></pre>	
<pre><luo:is_related_to luo:rel_type="luo:holonymy"></pre>	
<pre><luo:unit EWN:term_id="4345647" luo:type="Member" EWN:lang="ES"></pre>	
<pre><luo:holonymy>consejo</luo:holonymy></pre>	
<pre><luo:holonymy>guía</luo:holonymy></pre>	
<pre><luo:holonymy>orientación</luo:holonymy></pre>	
<pre></luo:unit></pre>	
<pre><luo:unit EWN:term_id="5292131" luo:type="Member" EWN:lang="ES"></pre>	
<pre><luo:holonymy>comité</luo:holonymy></pre>	
<pre><luo:holonymy>consejo</luo:holonymy></pre>	
<pre><luo:holonymy>junta</luo:holonymy></pre>	
<pre></luo:unit></pre>	
<pre></luo:is_related_to></pre>	
<pre></llo:semantics></pre>	
<pre></luo:word></pre>	
<pre></luo:token></pre>	

All along the annotated file, *XML namespaces* are used to refer to the ontological resource where a given linguistic unit, attribute or value comes from. Each **XML namespace**²¹⁰ implements the W3C-recommended mechanism for providing unique names to the elements and the attributes of an XML instance. The reason for doing so is that an XML instance may contain element or attribute names coming from more than one XML vocabulary. If each vocabulary is given a namespace then the ambiguity between identically named elements or attributes can be solved (provided that all element names within a namespace are unique). A namespace is declared using the reserved XML attribute *xmlns*, the value of which must be a URI (Uniform Resource Identifier)²¹¹ reference. For example, the item `<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">` defines the namespaces associated to the syntactic specification of RDF and RDF Schema, respectively.

The namespaces defined within OntoTagger, together with the ontological resource they identify (already described in previous subsections) are shown in Table 135.

Table 135: OntoTagger's namespaces

Namespace	Ontology
llo	Linguistic Level Ontology
luo	Linguistic Unit Ontology
lao	Linguistic Attribute Ontology
lvo	Linguistic Value Ontology
ldo	Linguistic Domain Ontology ²¹²
EWN	EuroWordNet
CNEO	Cinema Named Entities Ontology

The annotated example in Table 134 is analysed and explained next, in a dedicated table for each annotation level (or layer). Its **Lemma Tagging Layer** is analysed and explained first, in Table 136.

²¹⁰ http://en.wikipedia.org/wiki/XML_Namespace

²¹¹ A **Uniform Resource Identifier (URI)**, is a compact string of characters used to identify or name a resource. The main purpose of this identification is to enable interaction with representations of the resource over a network, typically the World Wide Web, using specific protocols. URIs are defined in schemes defining a specific syntax and associated protocols (http://en.wikipedia.org/wiki/Uniform_Resource_Identifier).

²¹² This ontology was implemented within OntoTag by reusing the domain (sub)classification included in SIMPLE (2000) and linking it to the LUO.

Table 136: Explanation of Lemma-related XML annotations within OntoTagger

XML ELEMENT	EXPLANATION
<pre data-bbox="148 481 429 593"><luo:token id="t:10_1_1" DataLexica_id="t:10_1_1" FDG_id="t:8_2_16" UMurcia_id="t:1_2_16"></pre>	<p data-bbox="445 271 1378 857">This is an open token label. This element stores the information of an instance of the simplest Syntactic Unit (a Token, which can be a Simple Token – consisting of just one word– or a MultiWord Token – in other case). Within the OntoTagger-annotated file, the token is identified by means of the value of its id attribute and characterised by the identifiers assigned to its DataLexica_id, FDG_id and UMurcia_id attributes. These last three identifiers point out to the corresponding token in the tool-specific annotation file obtained, before the combination sub-phase, for each tool incorporated into OntoTagger (respectively, Bitext’s DataLexica, Connexor’s FDG and LACELL’s POS tagger). The value of the id attribute for a given token is built by means of (1) a head letter, representing the unit it identifies (t, in this case), followed by a ‘:’ character, (2) its paragraph and sentence numbers and (3) the number of order of the token within the sentence it belongs to (separated by a ‘_’ character).</p>
<pre data-bbox="148 1008 347 1064"><luo:word id="w:10_1_1_1"></pre>	<p data-bbox="445 875 1378 1182">This is an open word label. It is the beginning of the annotations corresponding to a particular Word Form in the input text. Each word element is identified by means of the value of its id attribute. As with the token element, the value of the id attribute for a given word consists of (1) a head letter, representing the unit it identifies (w, in this case), followed by a ‘:’ character, (2) its paragraph, sentence and token numbers and (3) the number of order of the word within the token it belongs to (separated by a ‘_’ character).</p>
<pre data-bbox="148 1205 263 1279"><luo:text> Director </luo:text></pre>	<p data-bbox="445 1200 1378 1283">The element text is used to mark up the Word Form of the word, as it appears in the input file.</p>
<pre data-bbox="148 1323 300 1397"><luo:lemma> director </luo:lemma></pre>	<p data-bbox="445 1296 1378 1429">The element lemma labels the Lemma associated to the Word Form of the word in the input text, that is, the dictionary entry found when looking up the Word Form in a dictionary.</p>
<pre data-bbox="148 1447 277 1469"></luo:word></pre>	<p data-bbox="445 1442 1378 1473">Close word label. The annotation of a word ends with this label.</p>
<pre data-bbox="148 1491 285 1514"></luo:token></pre>	<p data-bbox="445 1487 1378 1518">Close token label. The annotation of a token ends with this label.</p>

Second, the morphosyntactic annotations (except for the Lemma Tagging Layer, described above) is analysed and explained in Table 137:

Table 137: Explanation of the morphosyntax-related XML annotations within OntoTagger

XML ELEMENT	EXPLANATION
<code><llo:morpho luo:category="luo:NC"></code>	This element opens the remaining morphosyntactic annotations of a token . The category attribute identifies the type of Morphosyntactic Unit (the grammatical category) being described. In this case, both the attribute and its NC value (<i>i.e.</i> , Common Noun) are extracted from the LUO, as the namespace preceding it (luo) states.
<code><lao:gender> lvo:M </lao:gender></code>	This annotation states that the morphological element gender is extracted from the lao and its values are taken from the lvo . In this case, its value, M (which is the Acronym of the Masculine concept of the LVO, associated to the Gender concept of the LAO), means that the word it is marking up has a Masculine Gender.
<code><lao:number> lvo:S </lao:number></code>	This annotation makes explicit that also the morphological element number is extracted from the lao and its values are taken from the lvo . In this case, the value S (which is the Acronym of the Singular concept of the LVO, associated to the Number concept of the LAO) means that the word it is marking up has a Singular Number.
<code><lao:case> lvo:N </lao:case></code>	Analogously, the morphological element case is extracted from the lao and its values are taken from the lvo as well. In this case, the value N (which is the Acronym of the Nominative concept of the LVO, associated to the Case concept of the LAO) means that the word it is marking up has a Nominative Case (the default Case value for Spanish words).
<code><lao:definiteness> 0 </lao:definiteness></code>	Also the morphological element definiteness is extracted from the lao and its values are taken from the lvo . In this case, the value 0 means that the linguistic attribute referenced by this element is not applicable to Spanish.
<code></llo:morpho></code>	Close morpho label. The morphosyntactic annotations of a token end with this label.

Third, the remaining syntactic annotations are explained in Table 138:

Table 138: Explanation of syntactic XML annotations within OntoTagger

XML ELEMENT	EXPLANATION
<code><llo:syntax luo:category="luo:TS"></code>	This element opens the syntactic annotation of a word or a token . The attribute category identifies the type of Syntactic Unit being described. In this case, the value TS (<i>i.e.</i> , the Acronym of the concept Simple Token in the LUO) shows that ‘Director’ is a token consisting of just one word . Note that (1) both the attribute category and its value, the Acronym TS, are preceded by the namespace luo , which refers to the ontology where they are defined, <i>i.e.</i> , the Linguistic Unit Ontology, and (2) the annotation level (syntax) is qualified by means of the namespace llo , associated to the Linguistic Level Ontology, which formalises the linguistic annotation levels.

XML ELEMENT	EXPLANATION
<pre><lao:surface_tag> lvo:N </lao:surface_tag></pre>	The surface_tag element is used to represent the annotations dealing with the Surface Tag attribute of the Linguistic Attribute Ontology (detailed by the namespace identifier preceding it, lvo). Its value (N) is the Acronym of the Nominal value included in the Linguistic Value Ontology (identified by its corresponding namespace, lvo). In this case, N means that the word in question is part of a Noun Phrase.
<pre><lao:phrase_function> lvo:H </lao:phrase_function></pre>	The description of the phrase_function element is analogous to the one of the surface_tag element. In this case, the value H (extracted from the LVO as well) shows that the word being annotated is the Head of the Phrase of the input text in which it is included. This attribute acquires its whole meaning in conjunction with the preceding one: they both, altogether, state that the present word is the Head of a Noun Phrase.
<pre><lao:morpho-syntactic_function> lvo:N </lao:morpho-syntactic_function></pre>	The morpho-syntactic_function element explains how the word is being used within a given context (participial verbs can act as adjectives, for example). In most cases, it is an underspecification of the morphosyntactic category of the word in question. In this case, N means that the word acts as a Noun within the Phrase where it appears.
<pre><luo:depends_syntactically_on head="w:9_1_2_1"> main </luo:depends_syntactically_on></pre>	The so-called depends_syntactically_on element implements the annotation of (functional) dependencies between syntactic units (words, in this case). The other unit on which the present one depends is identified by the value of the attribute head , which contains the identifier of the target word , w:9_1_2_1. The type of syntactic dependency holding between them is labelled by means of the text field of the element (main in this case, which indicates that the word governs the rest of the words in its Phrase ²¹³).
<pre></llo:syntax></pre>	Close syntax label. The syntactic annotation of a word or a token ends with this label.

Lastly, the **Semantic Level** annotations included in the example of Table 134 are described in detail in Table 139:

²¹³ Note that the type of syntactic dependency is not preceded by any namespace identifier. This is an inconsistency with respect to OntoTag's annotation scheme. It is due to the fact that, by the time when OntoTagger was built, syntactic dependencies were not conveniently represented yet within OntoTag's ontologies and, hence, within OntoTag's annotation scheme.

Table 139: Explanation of the semantic XML annotations within OntoTagger

XML ELEMENT	EXPLANATION
<pre><llo:semantics luo:category="luo:E"></pre>	<p>This element opens the <code>Semantic Level</code>-related tagging of a <code>word</code>. The attribute <code>category</code> identifies the type of Semantic Unit being described. Within <code>OntoTag(ger)</code>, only the four main types of semantic units are identified, namely <code>Entity</code> (E – the present case), <code>Process</code> (P), <code>Property</code> (R) and <code>Circumstance</code> (C). These types of units, as can be seen in the example, are extracted from the Linguistic Unit Ontology (denoted by the <code>luo</code> namespace).</p>
<pre><lao:use> lvo:[A P] </lao:use></pre>	<p>The element <code>use</code> represents the corresponding concept within the LAO (the <code>Use</code> attribute), which describes the way in which properties are attributed to entities (mainly by adjectives or adjectival phrases), either in an <code>Attributive</code> (A) or in a <code>Predicative</code> (P) way (included in <code>Ontotag</code>'s LVO). It is left underspecified in this case, since it is commonly used to qualify properties instead of entities (note the disjunction between both values, [A P], which constitutes an ambiguous –underspecified, in this case– annotation).</p>
<pre><luo:word_meaning></pre>	<p>Open <code>word_meaning</code> label. This element is used to mark up the start of the sense tagging of the <code>word</code>, performed by means of EuroWordNet (EWN) or any other suitable ontology (the Cinema Named Entity Ontology, CNEO, in this case) or linguistic resource.</p>
<pre><luo:word_sense luo:resource="CNEO" luo:term_id="100002#Director"/></pre>	<p>The element <code>word_sense</code> gives a concrete sense-tagging of the <code>word</code> being marked up. On the one hand, the attribute <code>resource</code> specifies the ontology according to which the <code>word</code> is sense-tagged (<i>i.e.</i>, the ontology where the sense tag is contained). In the present case, the <code>word_sense</code> has been extracted from the CNEO. On the other hand, the attribute <code>term_id</code> identifies the particular concept within the ontology that better describes the sense of the word being tagged. In the case of CNEO tags, the values of <code>term_id</code> are built out of the Name of the concept (<code>Director</code>) preceded by the value of its <code>Identifier</code> attribute within the ontology (100002 – see Section 5.4.3.2 for the details on how the values of <code>Identifier</code> are calculated). Both parts of the <code>term_id</code> values are separated by means of a '#' character.</p>
<pre><luo:word_sense luo:resource="EWN" luo:term_id="5921693"/></pre>	<p>In this case, the (ontological) <code>resource</code> according to which the <code>word</code> is sense-tagged is EuroWordNet (EWN), and the <code>term_id</code> attribute is assigned the value of (one of) the synset(s) where the <code>word</code> appears (5921693).</p>
<pre></luo:word_meaning></pre>	<p>Close <code>word_meaning</code> label. This element marks up the end of the sense-tagging of the word.</p>

XML ELEMENT	EXPLANATION
<pre><luo:is_related_to luo:rel_type="luo:synonymy"></pre>	The so-called <code>is_related_to</code> element opens the description of the semantic paradigmatic relationships of the <code>word_sense</code> with (an)other <code>word_sense(s)</code> within EuroWordNet. The type of semantic relationship is specified by means of the <code>rel_type</code> attribute (whose value is, in this case, <code>synonymy</code>). There can be several <code>is_related_to</code> elements associated to a <code>word</code> (to its <code>word_sense</code> , in fact), and even several of them relating to the same type of semantic relationship (<i>i.e.</i> , with the same <code>rel_type</code>).
<pre><luo:unit EWN:term_id="5921693" EWN:lang="ES"></pre>	The <code>unit</code> element, nested within the <code>is_related_to</code> element, details the sense to which the <code>word</code> in question (<i>i.e.</i> , its <code>word_sense</code>) relates (independently of the relationship holding between both senses). The target sense of the relation is specified by means of the value (5921693) of the attribute <code>term_id</code> . This value is a reference to a term in a certain (ontological) resource (EuroWordNet, in this case – note the namespace <code>EWN</code> qualifying it). The attribute <code>lang</code> is used to specify the particular language-dependent WordNet queried for this type of relationship (ES stands for Spanish, according to the ISO recommendations). There can be more than one <code>unit</code> element nested in an <code>is_related_to</code> element.
<pre><luo:synonymy> gerente </luo:synonymy></pre>	The element <code>synonymy</code> is used to list the words included within the EuroWordNet synset to which the <code>word_sense</code> in question is related by means of a <code>Synonymy</code> relationship. In this case, <code>gerente</code> is a <code>word</code> included in a synset (5921693) related to a synset where the <code>word</code> in question (<code>director</code>) appears.
<pre></luo:unit></pre>	Close <code>unit</code> element.
<pre></luo:is_related_to></pre>	Close <code>is_related_to</code> element.
<pre><luo:is_related_to luo:rel_type="luo:hyperonymy"></pre>	Another instance of the so-called <code>is_related_to</code> element; in this case, it opens the description of the <code>Hyperonymy</code> relationships of the <code>word_sense</code> .
<pre><luo:unit EWN:term_id="5921424" luo:type="null" EWN:lang="ES"></pre>	This is another example of use of the element <code>unit</code> . Its use is analogous to the case of <code>synonymy</code> , except for the possibility to subcategorise the <code>Hyperonymy</code> relationship, which can be attained by means of the <code>type</code> attribute (if no subcategorisation is specified, a <code>null</code> value is assigned to the attribute, as in this example)
<pre><luo:hyperonymy> administrador </luo:hyperonymy></pre>	The <code>hyperonymy</code> element is analogous to the <code>synonymy</code> element. It is used to list the words included within the EuroWordNet synset to which the <code>word_sense</code> in question is related by means of a <code>Hyperonymy</code> relationship.
<pre><luo:is_related_to luo:rel_type="luo:holonymy"></pre>	Yet another instance of the so-called <code>is_related_to</code> element. In this case, it opens the description of the <code>holonymy</code> relationships of the <code>word_sense</code> .
<pre><luo:unit EWN:term_id="4345647" luo:type="Member" EWN:lang="ES"></pre>	Another example of use of the element <code>unit</code> . Its use is analogous to the case of <code>synonymy</code> and <code>hyperonymy</code> (in this case, the <code>Member</code> value of the <code>type</code> attribute really subcategorises the kind of <code>Holonymy</code> relationship).

XML ELEMENT	EXPLANATION
<pre><luo:holonymy> consejo </luo:holonymy></pre>	The holonymy element is analogous to the synonymy and the hyperonymy elements. It is used to list the words included within the EuroWordNet synset to which the word_sense in question is related by means of a HoLonymy relationship.
<pre></llo:semantics></pre>	Close semantics label. The semantic annotation of a word ends with this label.

There remains a particular type of semantic annotation which is not included in the example shown in Table 134. This type of semantic annotation concerns the Instance Semantic Annotation Layer, *i.e.*, Named Entity annotation. A concrete example of this type of annotation is shown in Table 140 and described in Table 141.

Table 140: OntoTagger XML annotation of the named entity “Jesse James”

```
<luo:token id="t:6_1_4" DataLexica_id="t:6_1_4" FDG_id="t:8_1_7" UMurcia_id="t:1_1_9">
  <llo:discourse llo:category="luo:NE" domain="llo:cinema" subcategory="cneo:null" llo:MUC-7_tag="lvo:Enamex"/>
  <luo:word id="w:6_1_4_1">
    <luo:text>Jesse</luo:text>
    <luo:lemma>Jesse</luo:lemma>
    <llo:syntax luo:category="luo:TS">
      <lao:surface_tag>lvo:A</lao:surface_tag>
      <lao:phrase_function>lvo:R</lao:phrase_function>
      <lao:morpho-syntactic_function>lvo:N</lao:morpho-syntactic_function>
      <luo:depends_syntactically_on head="w:6_1_4_2">ada</luo:depends_syntactically_on>
    </llo:syntax>
    <llo:morpho luo:category="luo:NP">
      <lao:gender>lvo:[M|F|N]</lao:gender>
      <lao:number>lvo:S</lao:number>
      <lao:case>lvo:N</lao:case>
      <lao:definiteness>0</lao:definiteness>
    </llo:morpho>
  </luo:word>
  <luo:word id="w:6_1_4_2">
    <luo:text>James</luo:text>
    <luo:lemma>James</luo:lemma>
    <llo:syntax luo:category="luo:TS">
      <lao:surface_tag>lvo:N</lao:surface_tag>
      <lao:phrase_function>lvo:H</lao:phrase_function>
      <lao:morpho-syntactic_function>lvo:N</lao:morpho-syntactic_function>
      <luo:depends_syntactically_on head="w:6_1_3_1">mod</luo:depends_syntactically_on>
    </llo:syntax>
    <llo:morpho luo:category="luo:NP">
      <lao:gender>lvo:M</lao:gender>
      <lao:number>lvo:S</lao:number>
      <lao:case>lvo:N</lao:case>
      <lao:definiteness>0</lao:definiteness>
    </llo:morpho>
    <llo:semantics luo:category="luo:E">
      <luo:word_meaning>
        <luo:word_sense luo:resource="EWN" luo:term_id="5745149"/>
        <luo:word_sense luo:resource="EWN" luo:term_id="6452565"/>
        <luo:word_sense luo:resource="EWN" luo:term_id="6452640"/>
      </luo:word_meaning>
      <luo:is_related_to luo:rel_type="luo:synonymy">
        <luo:unit EWN:term_id="6452565" EWN:lang="ES">
          <luo:synonymy>Henry James</luo:synonymy>
        </luo:unit>
        <luo:unit EWN:term_id="6452640" EWN:lang="ES">
          <luo:synonymy>William James</luo:synonymy>
        </luo:unit>
      </luo:is_related_to>
    </llo:semantics>
  </luo:word>
</llo:discourse>
```

```

</luo:is_related_to>
<luo:is_related_to luo:rel_type="luo:hyperonymy">
  <luo:unit EWN:term_id="5724735" luo:type="null" EWN:lang="ES">
    <luo:hyperonymy>río</luo:hyperonymy>
  </luo:unit>
  <luo:unit EWN:term_id="6261110" luo:type="null" EWN:lang="ES">
    <luo:hyperonymy>filósofo</luo:hyperonymy>
  </luo:unit>
  <luo:unit EWN:term_id="6290582" luo:type="null" EWN:lang="ES">
    <luo:hyperonymy>psicólogo</luo:hyperonymy>
    <luo:hyperonymy>sicólogo</luo:hyperonymy>
  </luo:unit>
  <luo:unit EWN:term_id="6438760" luo:type="null" EWN:lang="ES">
    <luo:hyperonymy>autor</luo:hyperonymy>
    <luo:hyperonymy>escritor</luo:hyperonymy>
  </luo:unit>
</luo:is_related_to>
<luo:is_related_to luo:rel_type="luo:holonymy">
  <luo:unit EWN:term_id="5598208" luo:type="Part" EWN:lang="ES">
    <luo:holonymy>América</luo:holonymy>
    <luo:holonymy>E.E.U.U.</luo:holonymy>
    <luo:holonymy>Estados Unidos</luo:holonymy>
  </luo:unit>
</luo:is_related_to>
</llo:semantics>
</luo:word>
</luo:token>
    
```

As commented in the description of OntoTag's annotation scheme (Subsection 4.3) and in the description of the LUO (Subsection 4.1.3.1), named entities lie on the Semantics-Discourse interface and, accordingly, they could be annotated as units of any of these levels. Nevertheless, both their semantic and discourse-related features must be annotated, no matter in which of these two levels they are eventually included. As for OntoTagger's annotation schemas, named entities are considered units of both the `Discourse Level` and the `Semantic Level`. This allows for the separated annotation of their semantic and their discourse features and/or projections. This was achieved in OntoTagger's annotation schemas by means of the `(MultiWord) Token` unit. In fact, the different features of named entities are ascribed, according to their level, to the `Token` they constitute. Table 141 describes more concretely how this was implemented in these schemas.

Table 141: Description of the XML annotation of the named entity "Jesse James" in OntoTagger

XML ELEMENT	EXPLANATION
<code>llo:discourse</code>	This element makes explicit the level (note the Linguistic Level Ontology namespace, <code>llo</code>) to which the nested annotations pertain (<code>discourse</code> in this case).
<code>llo:category="luo:NE"</code>	The <code>category</code> attribute is used to identify the particular type of unit being annotated (within the corresponding level). In this case, it is stated that the type of unit is Named Entity (which is a concept of the LUO – note its namespace, <code>luo</code> , preceding the Acronym representing the unit, <i>i.e.</i> , <code>NE</code>).
<code>domain="llo:cinema"</code>	The meaning of the <code>domain</code> attribute is a rather obvious one: it allows for a kind of semantic-field tagging of the unit in question (which belongs to the <code>cinema</code> domain, in this case).

XML ELEMENT	EXPLANATION
<code>subcategory="cneo:null"</code>	The attribute <code>subcategory</code> details the subclassification (according to the <i>CNEO</i> ontology, note the <code>cneo</code> namespace preceding its value) obtained by the NERCA subsystem of OntoTagger for the Named Entity being annotated. In this case, the Named Entity could not be subclassified and, hence, the attribute was assigned a null value.
<code>lao:MUC-7_tag="lvo:Enamex"</code>	This attribute specifies the <code>tag</code> recommended in the <i>MUC-7</i> for the Named Entity being annotated. As can be deduced from their namespaces, <code>lao</code> and <code>lvo</code> , respectively, this attribute has been extracted from the Linguistic Attribute Ontology and its associated value (<code>Enamex</code> , standing for a Proper Named Entity, as opposed to, for example, <code>Timex</code> –time related– named entities) has been extracted from the Linguistic Value Ontology.

This completes the description of OntoTagger-XML-annotated documents. The format of OntoTagger-OWL-annotated documents is described in the following subsection.

5.5.2. OWL IMPLEMENTATION: AN EXAMPLE

The structure of OntoTagger-OWL-annotated documents is presented in this subsection. As with OntoTagger-XML-annotated documents, this section is based on a particular example of annotation as well (taken from the annotated counterpart of the so-called ODECORPUS-CINEMA corpus) for the sake of clarity and expressivity. This annotated example (corresponding to the Token associated to the Spanish Word Form ‘Director’) is distributed among Table 142, Table 143, Table 144 and Table 145. Table 142 shows the top-level description of the aforementioned annotation example, which refers to and links the annotations corresponding to each different Linguistic Level: the syntactic one (Table 143), the morphosyntactic one (Table 144), and the semantic one (Table 145). These level-specific annotations are instances of the Integration Ontology of OntoTag (the OIO). The elements in this schema are most similar to the ones included in OntoTagger-XML-annotated documents. Therefore, the explanations given here for these elements are mere descriptions of the particular way in which they have to be annotated in OWL.

As can be observed in Table 142, a `token`, which is a LUO unit (note the `luo` namespace qualifying it), is assigned the identifier `t:10_1_1` (by means of the RDF attribute `ID`). This identifier stands for “the first token of the first sentence in the tenth paragraph” (from right to left). It is using this identifier how the `token` in question can be referred to in the rest of the document and in other RDF or OWL documents (via XLink references, for example). In order to link this `token` with its corresponding tool-specific annotations within a tool-specific annotation file (for traceability, for example) some

particular attributes (namely `DataLexica_id`, `FDG_id` and `UMurcia_id`) have been included in the element `token` to contain the identifier of the `token` in each tool-specific annotation file.

Table 142: Example excerpt (1) from an OntoTagger-OWL-annotated document

```

<luo:token rdf:ID="t:10_1_1" DataLexica_id="t:10_1_1" FDG_id="t:8_2_16" UMurcia_id="t:1_2_16">
  <luo:word rdf:ID="w:10_1_1_1" luo:text="Director" luo:lemma="director">
    <llo:syntax>
      <luo:category rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++-+Integration#syntax_UAV-10_1_1_1"/>
    </llo:syntax>
    <llo:morpho>
      <luo:category rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++-+Integration#morpho_UAV-10_1_1_1"/>
    </llo:morpho>
    <llo:semantics>
      <luo:category rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++-+Integration#semantic_UAV-10_1_1_1"/>
    </llo:semantics>
  </luo:word>
</luo:token>

```

This example shows clearly the different way in which the tools integrated into OntoTagger segment the input file, since the `token` is:

- the sixteenth entry of the second `sentence` within the eighth `paragraph` for Connexor’s FDG (`FDG_id = "t:8_2_16"`)
- the sixteenth entry of the second `sentence` within the first `paragraph` for the LACELL’s POS tagger (`UMurcia_id = "t:1_2_16"`)
- the first entry of the first `sentence` within the tenth `paragraph` for Bitext’s DataLexica (`DataLexica_id = "t:10_1_1"`), which constitutes the preferred and combined (final) identifier for the `token`.

The following element of the example described in Table 142 is `word`. This element is a LUO unit as well, and it makes explicit the different components of the `token` in question. In this case, it is a Simple Token and, therefore, there is only a `word` element nested into the `token` element. In order for each `word` to have a convenient way to be referred, each one has been assigned an RDF attribute (`ID`) identifying it as well. The value of the `ID` attribute is built appending the order number of the `word` in question within the `token` it belongs to (the first one, in this case) to the token identifier (`10_1_1`); the result, in this case, is the value `"w:10_1_1_1"` assigned to the `rdf:ID` attribute of `word`. Besides, each `word` element contains two other attributes: `luo:text` and `luo:lemma`, which are used to attach to the `word` element, respectively, its corresponding Word Form in the input file and its Lemma (determined in the annotation process).

Apart from the `lemma`, also the remaining annotations at the morphosyntactic (`llo:morpho`) and annotations at the syntactic (`llo:syntax`) and the Semantic Level (`llo:semantics`) are included, for each `token` in the input file, within OntoTagger-OWL-annotated documents. Note that each type of

annotation is preceded by the namespace defined for OntoTag's Linguistic Level Ontology (llo), where these levels are defined as concepts. These level-specific annotations are instances of OntoTag's Integration Ontology (OIO) and are identified uniquely within this ontology via the concatenation of their level and their **word** identifier: respectively, `morpho_UAV-10_1_1_1`, `syntax_UAV-10_1_1_1` and `semantic_UAV-10_1_1_1`. As mentioned above, the OWL code describing each of these instances is shown in Table 143, Table 144 and Table 145 (in their generation order, *i.e.*, syntax, morphosyntax and semantics). Each of these tables is described in detail below.

The **annotations at the Syntactic Level** are instanced by means of the OWL code in Table 143. This table shows the instance of the `Unit` concept of the OIO identified by the string `syntax_UAV-10_1_1_1`, which is accompanied by its corresponding instances of the OIO concepts `Attribute` and `Value`. In other words, this string stands for the set of instances of the `<Unit, Attribute, Value>` Triple annotating the **word** in question (which is identified by the string `10_1_1_1`) at the `Syntactic Level`.

Table 143: Example excerpt (2) from an OntoTagger-OWL-annotated document: a Syntactic Level instance

```

<Unit rdf:ID="syntax_UAV-10_1_1_1">
  <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++Linguistic+Units+2#Simple Token"/>
  <Has>
    <Attribute>
      <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++Linguistic+Attributes+2#Surface
Tag"/>
      <Takes>
        <Value>
          <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++
+Linguistic+Values+2#Nominal"/>
          </Value>
        </Takes>
      </Attribute>
      <Attribute>
        <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++Linguistic+Attributes+2#Phrase
Function"/>
        <Takes>
          <Value>
            <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++
+Linguistic+Values+2#Head-function"/>
            </Value>
          </Takes>
        </Attribute>
      <Attribute>
        <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++Linguistic+Attributes+2#Morpho-
Syntactic Function"/>
        <Takes>
          <Value>
            <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++
+Linguistic+Values+2#Noun"/>
            </Value>
          </Takes>
        </Attribute>
      <Attribute>
        <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag++
+Linguistic+Units+2#Depends_Syntactically_On"/>
        <Takes>
          <Value>
            <rdf:Value head="w:9_1_2_1" rdf:resource="main"/>
            </Value>
          </Takes>
      </Attribute>
    </Has>
  </Unit>

```

```

</Attribute>
  </Has>
</Unit>

```

The **word** annotated in the example is the only component of a Simple Token and, therefore, it is annotated as such, via the **resource** RDF attribute in the example, which states that it is a Simple Token, which is a type of Unit within the LUO (identified in the example by the string “OntoTag – Linguistic Units 2” –*i.e.*, its second release– within the XLink-fashion value assigned to the aforementioned **resource** attribute).

The **word** in question has been annotated with several syntactic attributes and values, which are nested between the **<Has>** and the **</Has>** XML labels in the example. These labels are used to describe the instances of the Has relationship of the OIO, which holds between a Unit and an Attribute. The Unit described in the example has been associated four attributes that characterise it. On the one hand, the Surface Tag, Phrase Function and Morpho-Syntactic Function attributes represent the corresponding syntactic attributes defined in the LAO (identified in the example by the string “OntoTag – Linguistic Attributes 2”). On the other hand, the attribute Depends_Syntactically_On represents its corresponding syntactic relationship of the LUO.

Analogously, the linguistic values assigned to each linguistic attribute in the annotation are listed in the OWL document by means of the XML labels **<Takes>** and **</Takes>**. These labels are used to describe the instances of the Takes relationship of the OIO, which holds between an Attribute and a Value. The concrete values for the attributes mentioned above are defined in the LVO (identified in the example by the string “OntoTag – Linguistic Values 2”). A detailed description of the actual meaning of each attribute and value included in the annotation can be found in Sections 4.3 and 5.5.1.

Note that namespaces are used in the example (and all throughout the annotation schema) only to refer to the elements and the attributes that are not defined in the OIO, *i.e.*, for RDF elements and attributes. The rest of the elements and attributes used in the annotation schema are implicitly assigned the OIO (OntoTag’s Integration Ontology) namespace, where they are defined. This example shows the importance of using namespaces: should they not be used, the **Value** element, defined in the LUO, would be ambiguous with respect to the **rdf:Value** element (defined in RDF). In RDF, the **Value** element is treated as the concrete value that is assigned to its parent element in the tree structure representing the RDF / OWL document where it appears (*e.g.*, the **Value** –the type– of Unit is Simple Token in the example).

The **morphosyntactic annotations** are instanced by means of the OWL code included in Table 144. This table shows the instance of the Unit concept of the OIO identified by the string

morpho_UAV-10_1_1_1, which is accompanied by its corresponding instances of the OIO concepts *Attribute* and *Value*. In other words, this string stands for the set of morphosyntactic instances of the $\langle \text{Unit}, \text{Attribute}, \text{Value} \rangle$ Triple annotating the **word** in question (which is identified by the string 10_1_1_1).

Table 144: Example excerpt (3) from an OntoTagger-OWL-annotated document: a morphosyntactic annotation instance

```

<Unit rdf:ID="morpho_UAV-10_1_1_1">
  <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+-+Linguistic+Units+2#Common Noun"/>
  <Has>
    <Attribute>
      <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+-+Linguistic+Attributes+2#Gender"/>
      <Takes>
        <Value>
          <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+-+Linguistic+Values+2#Masculine"/>
        </Value>
      </Takes>
    </Attribute>
    <Attribute>
      <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+-+Linguistic+Attributes+2#Number"/>
      <Takes>
        <Value>
          <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+-+Linguistic+Values+2#Singular"/>
        </Value>
      </Takes>
    </Attribute>
    <Attribute>
      <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+-+Linguistic+Attributes+2#Case"/>
      <Takes>
        <Value>
          <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+-+Linguistic+Values+2#Nominative"/>
        </Value>
      </Takes>
    </Attribute>
    <Attribute>
      <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+-+Linguistic+Attributes+2#Definiteness"/>
      <Takes>
        <Value>
          <rdf:Value rdf:resource="0"/>
        </Value>
      </Takes>
    </Attribute>
  </Has>
</Unit>

```

As with syntactic units, this Morphosyntactic Unit has been associated an *rdf:Value* attribute, which makes explicit the actual type of Morphosyntactic Unit (or category) it is. In this case, it is a Common Noun, and it is annotated as such in the OWL file via the *resource* RDF attribute, which states that that it is a Common Noun – its corresponding Unit within the LUO (identified in the example by the string “OntoTag – Linguistic Units 2”).

The **word** in question has been annotated with several morphosyntactic attributes and values, which are nested between the `<Has>` and the `</Has>` XML labels, as with syntactic units. The `Unit` described in the example has four attributes characterising it: Gender, Number, Case and Definiteness, which represent their associated syntactic attributes of the LAO (identified in the example by the string “OntoTag – Linguistic Attributes 2”).

Analogously, the linguistic values assigned to each linguistic attribute in the annotation are listed in the OWL document by means of the `<Takes>` and `</Takes>` XML labels. The concrete values for the attributes aforementioned are, respectively, Masculine, Singular, Nominative and 0 (Not Applicable). Each of them (except for 0) is defined via its associated concept of the LVO (identified in the example by the string “OntoTag – Linguistic Values 2”). A detailed description of the actual meaning of each attribute and value included in the annotations at this level can be found in Sections 4.3 and 5.5.1.

The **annotations at the Semantic Level** are instanced by means of the OWL code in Table 145 (see next page). This table shows the instance of the `Unit` concept of the OIO identified by the string `semantic_UAV-10_1_1_1`, which is accompanied by its corresponding instances of the OIO concepts `Attribute` and `Value`. In other words, this string stands for the set of instances of the `<Unit, Attribute, Value>` Triple annotating the **word** in question (which is identified by the string `10_1_1_1`) at the `Semantic Level`. For the sake of space, it has not been included all the semantic annotation concerning the `Unit` in question within the table. In the actual OWL file, there appears some semantic annotations about the `Hyperonymy` and `Holonymy` relationships of the `Unit` with other units. However, the way in which they are annotated in the OWL document is most similar (in format and content) to the way in which `Synonymy` relationships are annotated. Therefore, only the annotation of this last relationship has been excerpted and included in Table 145 (and described here).

As with syntactic and morphosyntactic units, this `Semantic Unit` has been associated an `rdf:Value` attribute, which makes explicit the actual type of `Semantic Unit` it is. In this case, it is an `Entity`, and it is annotated as such in the OWL file via the `resource` RDF attribute, which states that that it is an `Entity`, its corresponding `Unit` within the `LUO`.

Table 145: Example excerpt (4) from an OntoTagger-OWL-annotated document: a `Semantic Level` instance

```

<Unit rdf:ID="semantic_UAV-10_1_1_1">
  <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag--+Linguistic+Units+2#Entity"/>
  <Has>
    <Attribute>
      <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+
+Linguistic+Attributes+2#Use"/>
    <Takes>
      <Value>

```

```

        <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+
        +Linguistic+Values+2#Attributive"/>
    </Value>
    <Value>
        <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+
        +Linguistic+Values+2#Predicative"/>
    </Value>
</Takes>
</Attribute>
<Word_Meaning>
    <Word_Sense>
        <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+
        +Cinema+Named+Entities#100002#Director"/>
    </Word_Sense>
    <Word_Sense>
        <rdf:Value rdf:resource="EWN#5921693"/>
    </Word_Sense>
    ...
</Word_Meaning>
<Is_Related_To>
    <Rel_Type>
        <rdf:Value rdf:resource="webode://webode.dia.fi.upm.es:2000/OntoTag+
        +Linguistic+Units+2#luo:synonymy"/>
    <Rel_Unit>
        <Term_Id>
            <rdf:Value rdf:resource="EWN#5921693"/>
        </Term_Id>
        <Lang>
            <rdf:Value rdf:resource="EWN#ES"/>
        </Lang>
        <synonymy>
            <rdf:Value rdf:resource="EWN#gerente"/>
        </synonymy>
        <synonymy>
            <rdf:Value rdf:resource="EWN#manager"/>
        </synonymy>
    </Rel_Unit>
    <Rel_Unit>
        <Term_Id>
            <rdf:Value rdf:resource="EWN#6013928"/>
        </Term_Id>
        <Lang>
            <rdf:Value rdf:resource="EWN#ES"/>
        </Lang>
        <synonymy>
            <rdf:Value rdf:resource="EWN#director de orquesta"/>
        </synonymy>
    </Rel_Unit>
    ...
</Rel_Type>
</Is_Related_To>
<Is_Related_To>
    ...
</Is_Related_To>
</Has>
</Unit>

```

As with syntactic and morphosyntactic units, this Semantic Unit has been associated an `rdf:Value` attribute, which makes explicit the actual type of Semantic Unit it is. In this case, it is an Entity, and it is annotated as such in the OWL file via the `resource` RDF attribute, which states that that it is an Entity, its corresponding Unit within the LUO.

The `word` in question has only one attribute to be annotated, and its annotation is nested between the `<Has>` and the `</Has>` XML labels, as with the syntactic and morphosyntactic units. This

annotation corresponds to the attribute `Use`, which represents the associated `Semantic Attribute` of the LAO. The other two main elements nested between the `<Has>` and the `</Has>` XML labels are `Word_Meaning` and `Is_Related_To`, which contain the annotations of the semantic relationships of the unit in question with its sense (according to EuroWordNet synsets or any other (ontological) resource) and with other semantic units (such as synonymy, hyperonymy, etc.), respectively.

Analogously, the linguistic values assigned to the `Use Semantic Attribute` in this annotation example are listed in the OWL document between the `<Takes>` and `</Takes>` XML labels. Therefore, the concrete values assigned to the attribute aforementioned are `Attributive` and `Predicative`. Each of them is defined via its associated concept of the LVO. Note that, in this case, two values have been assigned concurrently to the same attribute. This might constitute an ambiguous annotation of the referred attribute. However, in this case, it constitutes only an underspecification of the value of this attribute, since these two values are the only ones that it can take. In fact, as mentioned above, this attribute is supposed to be applied to properties (not to entities). A detailed description of the actual meaning of each attribute and value included in the annotations at this level can be found in Sections 4.3 and 5.5.1.

A similar description to the one given above could have been included here for the annotation of named entities in an OWL format within OntoTagger. It has not been included, though, in order to avoid redundancy. Thus, this concludes the presentation of both the annotation configuration and the annotation schemas of OntoTagger, developed for the evaluation of the OntoTag model. The results of this evaluation are shown in the following section.

6. RESULTS AND EVALUATION

As commented, OntoTagger is an automatic system developed with a threefold aim: (i) supporting (or refuting) OntoTag's underlying hypotheses; (ii) verifying and validating in practice OntoTag's abstract annotation scheme and architecture with respect to the objectives presented in Chapter 2 (Work Objectives, see page19); and (iii) assessing somehow the validity and suitability of the model.

In particular, on the one hand, *the architecture configuration of OntoTagger was implemented in order to find out*: (a) whether the separate results of linguistic annotation tools can interoperate and be united together in a sort of combined linguistic annotation, with more accurate results than any of the initial (separate) annotation results (or not); and, if they can interoperate and be combined, (b) how this combination should be performed, by means of a concrete example; as well as (c) what the problems that arise during this combination are, and how they could be solved.

On the other hand, *the annotation schemas of OntoTagger were implemented in order to determine whether (or not) ontologies and Semantic Web languages are suitable for the implementation of (a) standardised and/or standard-compliant linguistic annotation schemas, and (b) the intercommunication mechanisms for (linguistic) tools to interoperate and elaborate, collaboratively, multi-layered (or, equivalently, multi-level) linguistically-annotated corpora.*

As for the **evaluation of the goals of the three OntoTagger's annotation schemas**, firstly, it was meant to incorporate into OntoTagger as many (official or *de facto*) linguistic standards as possible. This was achieved by means of the formalisation, within OntoTag's ontologies, of the standards that existed already when the system was being developed. This included EAGLES (1996a; 1996b) recommendations, as well as the ISO TC 37 SC4 standards (such as ISO/LAF (2009) – all of them under development). ISO/LAF-related criteria (such as readability or extensibility) were fulfilled both with (i) the formalisation, in OntoTag's ontologies, of the linguistic categories (or units) and features (or attributes and values) used in linguistic annotation, and (ii) their subsequent use, in the annotation schema, in conjunction with the Semantic Web standard languages. And secondly, each of them (that is, the XML-, the RDF(S)- and the OWL-based) shows that any of these standard languages is suitable for the elaboration of standard-compliant annotation schemas with respect to their implementation format and/or language. In addition, it was also shown, by means of a concrete example, how linguistic annotations can be implemented by means of a Semantic Web language.

As for the **evaluation of the goals related to OntoTagger's architecture configuration**, it is based mainly on the different results presented in the following two subsections (and their embedded subsections). The first one (Subsection 6.1) includes some comparative statistics of the results of each separate tool integrated into the configuration of OntoTagger, together with the results obtained by

combining the former ones in the configuration. This comparative statistics contribute highly to a positive evaluation of these goals, and visibly support the underlying hypotheses of OntoTag associated to its combination sub-phase. The second one (Subsection 6.2) shows some preliminary results corresponding to the Named Entity Recognition, Classification and Annotation (NERCA) module of OntoTagger mentioned above, which also contribute to evaluate the achievement of the aforementioned goals.

6.1. EVALUATION OF THE COMBINATION SUB-PHASE

As previously mentioned, the performance of the different external tools incorporated into OntoTagger was characterised by some statistic measures, derived from their output annotations, and compared to the analogous statistic measures characterising the performance of OntoTagger itself, derived from its output (combined) annotations. This comparison aimed at: (i) refining and improving the combination sub-phases included in the aforementioned implementation of OntoTag, (ii) validating (eventually) the aforementioned sub-phases, which entailed (iii) corroborating the corresponding OntoTag hypothesis, that is, that the separate results of linguistic annotation tools can be united together in a sort of combined linguistic annotation, with more accurate results than any of the initial (separate) annotation results.

The methodology followed for the generation of these comparative statistic measures can be described as follows: firstly, a dedicated corpus was compiled, namely the ODECORPUS-Entertainment. This corpus was created by the linguistic researchers of the OEG research group²¹⁴. It was developed in the context of the project ContentWeb, within which OntoTagger was developed as well. It consists mainly of web pages on film and theatre play reviews, as well as restaurant commentaries, written in Spanish. The foremost sources considered were la ‘Guía del ocio’ (<http://www.guiadelocio.es>) and ‘La Netro’ (<http://www.lanetro.com>).

Secondly, a small sample²¹⁵ (that is, the first ten pages or, equivalently, around 5000 morphosyntactic units) of the aforementioned corpus was (were) morphologically and morphosyntactically annotated by means of the three external tools incorporated into the configuration of OntoTagger, namely LACELL’s tagger, Connexor’s FDG and Bitext’s DataLexica (henceforth referred to as UMurcia, FDG and DataLexica, respectively).

²¹⁴ <http://www.oeg-upm.net>.

²¹⁵ The scope of the experiments carried out had to be reduced to a minimum, due to time and funding constraints. Besides, the LACELL’s tagger was not actually available for its integration into OntoTagger. However, the LACELL research group of the Universidad de Murcia kindly provided the OEG with the morphosyntactic annotation of the aforementioned sample of the ODECORPUS-Entertainment, which needed to be incorporated as well into the present study.

Thirdly, as explained in Subsection 5.3 – the combination module, a particular (temporary or partial) combination XML document was generated by each OntoTagger combination sub-phase, in order for its results to be compared with those of each of the external tools.

Fourthly, the results obtained for the small sample of the ODECORPUS-Entertainment aforementioned were manually checked and corrected, and a manually-corrected XML document for each combination sub-phase was generated as well.

Fifth, all throughout the process of refinement of the morphosyntactic²¹⁶ combination heuristics, this manually-corrected XML documents were used as a gold standard to calculate several different statistical indicators, detailed below, intended to assess the degree of precision and of recall the results provided by the four annotation systems involved (namely the three external annotation tools, plus OntoTagger) in each of these sub-phases of the combination process.

In what follows, we present two different types of statistical indicators. On the one hand, there are statistical indicators which pertain to a certain sub-phase of OntoTagger. On the other hand, there are generic statistical indicators which can be defined similarly for every OntoTagger combination sub-phase. **Whereas the particular statistical indicators used in particular sub-phases give an idea of the recall of the corresponding annotations, the generic statistical indicators give an idea of the precision of the corresponding annotations.** The particular statistical indicators used in particular sub-phases will be introduced in the corresponding sub-phase; on the contrary, the generic indicators will be presented here for the sake of conciseness.

Hence, the **generic statistical indicators** used in every combination sub-phase took into account two main criteria for the characterisation of the particular annotation element under study in each combination sub-phase (henceforth referred to as an item). The first criterion characterised the *precision* of the item with respect to the gold standard. The second criterion characterised the item with respect to the *degree of disagreement* among the different external tools when annotating it. Accordingly, the combination of these two criteria generated four different (and complementary) statistical indicators for each of the tools involved: (1) the average number of items correctly annotated by the corresponding tool in a context with full annotation agreement; (2) the average number of items incorrectly annotated by the corresponding tool in a context with full annotation agreement; (3) the average number of items correctly annotated by the corresponding tool in a context with some annotation disagreement; and (4) the average number of items incorrectly correctly annotated by the corresponding tool in a context with some annotation disagreement.

²¹⁶ The annotations of the three linguistic tools integrated into OntoTagger overlapped only for morphosyntactic phenomena.

Thus, the different results obtained eventually for every combination sub-phase are presented next, each in a dedicated subsection, together with their corresponding comparative statistical indicators. First, in Subsection 6.1.1, it is included the results of the combination of morphosyntactic categories within OntoTagger; second, it will be shown, in Subsection 6.1.2, the results of the Lemma combination sub-phase; and third, it will be presented the results associated to the sub-phase for morphological attribute combination (in Subsection 6.1.3).

6.1.1. STATISTICAL ANALYSIS OF THE MORPHOSYNTACTIC CATEGORY COMBINATION RESULTS

In the present subsection it is shown the comparative results used for the evaluation of the morphosyntactic combination sub-phase of OntoTagger. First, the results for the generic statistical indicators are shown in Subsection 6.1.1.1; then, it will be presented the particular statistical indicators associated to this sub-phase, together with their corresponding (comparative) results (in Subsection 6.1.1.2).

6.1.1.1 PRECISION-RELATED (GENERIC) STATISTICAL INDICATORS

As already stated, it is presented in this subsection the particular statistical indicators of the morphosyntactic category combination sub-phase. They give an idea of the precision of the morphosyntactic category tag associated, by each of the tools considered (namely UMurcia, FDG, DataLexica and OntoTagger), to the different morphosyntactic units of the input text. The values of the generic indicators associated to this combination sub-phase are shown in Figure 26 (below).

The bar chart included in Figure 26 presents the statistical indicators associated to the morphosyntactic category tags assigned to the items in the input text (loosely referred to as *words*). The results of each statistical indicator, namely *Words without disagreements with correct result*, *Words without disagreements with wrong result*, *Words with disagreements with correct result*, and *Words with disagreements with wrong result*, have been calculated as explained above. They have been grouped together in order to show more clearly the behaviour of each tool, namely OntoTagger (in light blue), DataLexica (in dark blue), FDG (in maroon), and UMurcia (associated to LACELL's tagger, in beige), with respect to the other ones.

As for the first group of indicators in Figure 26, referred to as *Words without disagreements with correct result*, it shows the average number of morphosyntactic units in the input text that were annotated correctly by each tool. This entails that the corresponding tag coincided with the one assigned to the same item in the gold standard (*i.e.*, the corpus sample annotated and manually

corrected beforehand). As commented above, this statistical indicator takes into account that all the tools assigned the same morphosyntactic tag to the morphosyntactic unit in question (that is, the tags were combined *without disagreements*).

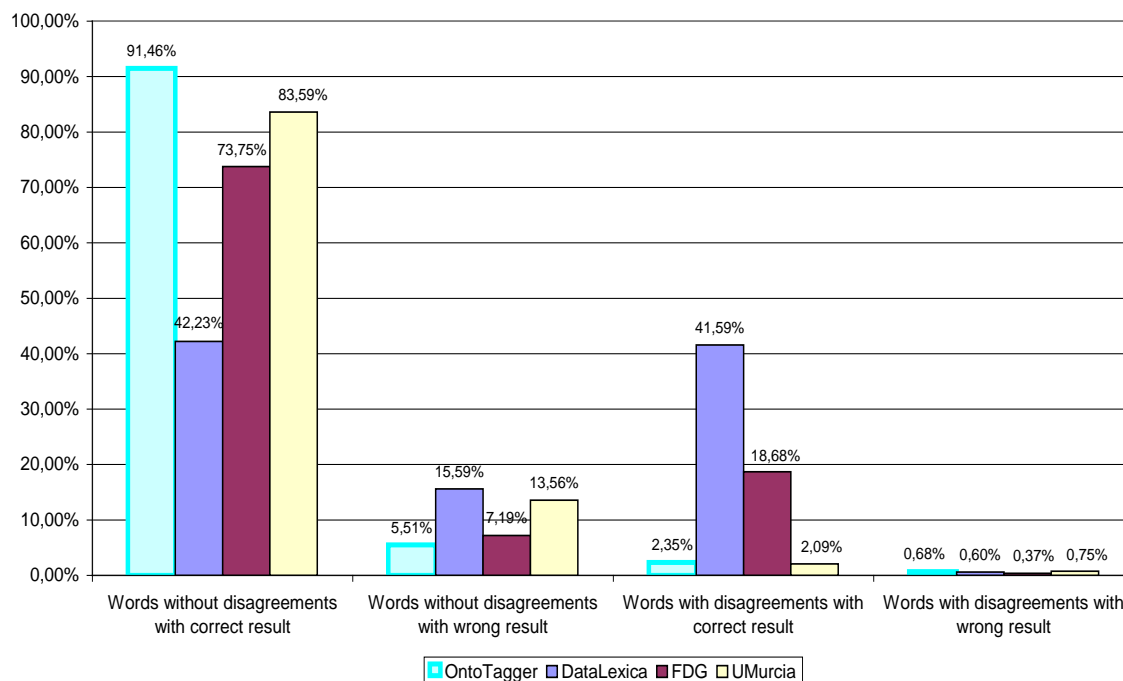


Figure 26: Comparative statistics associated to the precision of morphosyntactic categories

As can be seen in the aforementioned figure, the value of this particular indicator for OntoTagger (91.46%) highly outperform those of DataLexica (42.23%), improve significantly those of UMurcia (73.75% – OntoTagger is more precise in around the 18% of these cases), and clearly surpass the results of FDG (whereas the latter yields a value of 83.59% for this indicator, the former yields a 91.46%, which indicates that OntoTagger outperforms FDG in around the 8% of these cases).

As for the second group of indicators in Figure 26, referred to as *Words without disagreements with wrong result*, it shows the average number of morphosyntactic units in the input text that were annotated incorrectly by each tool. This entails that the corresponding tag did not coincide with the one assigned to the same item in the gold standard. As commented above, this statistical indicator takes into account that all the tools assigned the same morphosyntactic tag to the morphosyntactic unit in question (that is, the tags were combined *without disagreements*).

As can be seen in the aforementioned figure, the value of this particular indicator for OntoTagger (5.51%) outperforms those of the other tools as well. Whereas a high value of the previous indicator entails a higher number of correctly annotated tags and, hence, points out a good performance of the tool under consideration, a high value of the present indicator points out a bad performance, since it entails a higher number of wrongly annotated items. Accordingly, OntoTagger results outperform

three times those of DataLexica (15.59%), and twice those of UMurcia (13.56%); they also surpass slightly the results of FDG (7.19%).

As for the third group of indicators in Figure 26, referred to as *Words with disagreements with correct result*, it also shows the average number of morphosyntactic units in the input text that were annotated correctly by each tool. However, in this case, this indicator summarises the results for the cases in which not all the tools assigned the same morphosyntactic tag to the morphosyntactic unit in question (that is, the tags were combined *with disagreements*).

As can be seen also in the aforementioned figure, the value of this particular indicator for DataLexica (41.59%), highly outperforms those of FDG (18.68%), UMurcia (2.09%), and OntoTagger (2.35%). This is due to the fact that DataLexica is a morphological and non-disambiguating annotating tool, that is, it gives all the possible morphosyntactic tags for each item in the input text. Hence, only in a very small set of cases it gives an incorrect result. However, the correct tag has to be conveniently discerned from the (sometimes exceedingly numerous) ones, and, unfortunately, this task is not performed by DataLexica.

Finally, as for the fourth group of indicators in Figure 26, referred to as *Words with disagreements with wrong result*, it shows the average number of morphosyntactic units in the input text that were annotated incorrectly by each tool. Also in this case, this indicator summarises the results for the cases in which not all the tools assigned the same morphosyntactic tag to the morphosyntactic unit in question (that is, the tags were combined *with disagreements*).

As can be seen in the aforementioned figure, the value of this particular indicator for each and every tool are similar and good, since all of them are below 1% (OntoTagger = 0.68%; DataLexica = 0.60%; FDG = 0.37%; and UMurcia = 0.75%). Accordingly, in this case, this indicator cannot be considered significant.

So, according to the results presented in the current subsection, especially those concerning the first and the second indicators, it can be considered that OntoTagger outperforms the rest of the tools. Hence, up to this point, they support the aforementioned underlying hypothesis, and also allow for a positive evaluation of the corresponding objectives of OntoTag.

6.1.1.2 RECALL-RELATED (PARTICULAR) STATISTICAL INDICATORS

As already commented, this subsection presents the particular statistical indicators of the morphosyntactic category combination sub-phase. They give an idea of the recall of the

morphosyntactic category tag assigned, by each of the tools considered (namely UMurcia, FDG, DataLexica and OntoTagger), to the different morphosyntactic units of the input text.

Two different kinds of particular statistical indicators were devised for evaluating the present combination sub-phase of OntoTagger. The first group of these indicators is calculated simply as the difference in the average number of input items which are assigned a more specific morphosyntactic tag by the pair of tools being compared. For this purpose, for instance, the tags “NC” (Noun, Common) and “NP” (Noun, Proper) should be regarded as more specific than “N” (Noun), and the tag “RU” (Residual, Unclassified) should be regarded as the least specific one.

Consequently, this first group of indicators is calculated as follows: first, for each pair of tools, i and j , where i and j can be *OntoTagger*, *FDG*, *DataLexica* or *UMurcia*, we computed the set of input items whose corresponding morphosyntactic category tags do not coincide (referred to as $S_{disagreement}$). Second, for each tool, we obtained the corresponding subset of $S_{disagreement}$ whose members are the input items for which the tool i gives a correct and most accurate tag of the two tools of the pair (referred to as $S_{i\text{moreAccurate}}$). Third, let $|S_{disagreement}|$ and $|S_{i\text{moreAccurate}}|$ be the cardinalities of $S_{disagreement}$ and $S_{i\text{moreAccurate}}$, respectively. Then, for each of the two tools, we determined the value $ratio_i$, which results from dividing $|S_{disagreement}|$ by $|S_{i\text{moreAccurate}}|$. Finally, the indicator in question is calculated as $ratio_i - ratio_j$. Obviously, whereas a positive value of this indicator means that the tool i outperforms the tool j , a negative value of this indicator means that the tool j outperforms the tool i . When this indicator equals 0.00%, this means that the two tools being compared are equivalently accurate.

As far as the second group of these indicators is concerned, it further characterises the first one. Indeed, it measures the average number of items which are attached a more specific tag by a given tool than the other(s), but only in some particular cases. These particular cases are the ones in which the different tools disagree only partially. More concretely, in these cases, the tools do not agree in the assignment of the specific part of the morphosyntactic tag but, on the contrary, they do agree in the assignment of its higher-level part. The higher-level part of a morphosyntactic tag is the one associated to the sub-classification of the morphosyntactic unit into one of the major word categories, which is the only attribute considered obligatory in the EAGLES (1996a) recommendations for morphosyntactic annotation. Its corresponding values are represented by: “N”, “V”, “AJ”, “PD”, “AT”, “AV”, “AP”, “C”, “NU”, “I”, “U”, “R” and “PU”²¹⁷. The specific part of the morphosyntactic tag would be the remaining part. For example, the morphosyntactic categories associated to “NC” (Noun, Common) and “NP” (Noun, Proper) fall into this type of cases, since they share the higher-

²¹⁷ Which, respectively, stand for the morphosyntactic categories: Noun, Verb, Adjective, Pronoun/Determiner, Article, Adverb, Adposition, Conjunction, Numeral, Interjection, Unique/unassigned, Residual and Punctuation mark.

level part of their morphosyntactic tag (“N” – Noun), but not their most specific parts (respectively, (“C” – Common, and “P” – Proper).

Accordingly, this second group of indicators is calculated as follows: first, for each pair of tools, i and j , where i and j can be *OntoTagger*, *FDG*, *DataLexica* or *UMurcia*, we computed the set of input items whose corresponding morphosyntactic category tags do not coincide (referred to as $S_{disagreement}$). Second, we obtained the subset of $S_{disagreement}$ for which the higher-level part of the corresponding (correct) morphosyntactic tags coincide (referred to as $S_{partialMatch}$). Third, for each tool, we obtained as well the corresponding subset of $S_{partialMatch}$ whose members are the input items for which the tool i gives the most accurate tag of the two tools of the pair (referred to as $S_{i\ more\ Accurate}$). Fourth, let $|S_{partialMatch}|$ and $|S_{i\ more\ Accurate}|$ be the cardinalities of $S_{partialMatch}$ and $S_{i\ more\ Accurate}$, respectively. Then, for each of the two tools, we determined the value $ratio_i$, which results from dividing $|S_{partialMatch}|$ by $|S_{i\ more\ Accurate}|$. Finally, we calculated the indicator in question as $ratio_i - ratio_j$. In this case, as well, a positive value of this indicator means that the tool i outperforms the tool j , whereas a negative value of this indicator means that the tool j outperforms the tool i . As in the previous case, when this indicator equals 0.00%, this means that the two tools being compared are equivalently accurate.

Hence, we computed, according to the algorithms described above, the resulting values of these two kinds of particular indicators associated to this combination sub-phase. They are shown in the bar chart of Figure 27. Whereas the values associated to the first group are included on the left part of the bar chart, those corresponding to the second group have been included on its right part.

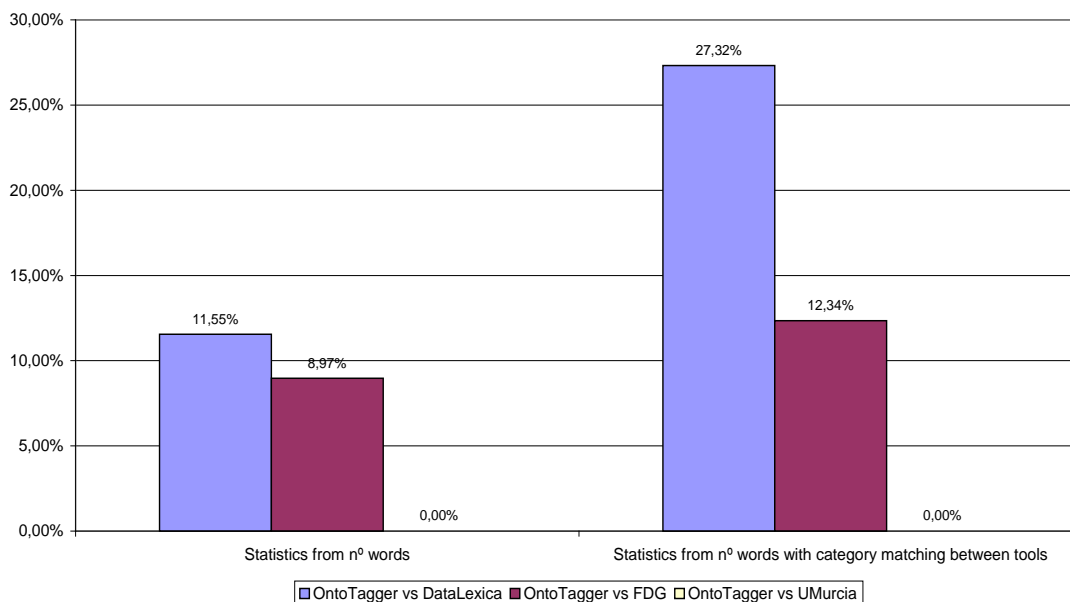


Figure 27: Comparative statistics associated to the recall of morphosyntactic categories

Regarding the values of the indicators in the first group, they are referred to as *Statistics from n° words* in Figure 27. As can be seen in the aforementioned figure, OntoTagger clearly outperforms DataLexica in the 11.55% of the cases (the respective value of this comparative indicator is shown in blue in the bar chart), and FDG in the 8.57% of the cases (the respective value of this comparative indicator is shown in maroon in the bar chart). However, the third value of this comparative indicator shows that OntoTagger and UMurcia are equivalently accurate. This is due to the fact that, in fact, the UMurcia morphosyntactic tags, when correct, are the most accurate of the three outputted by the three external tools. Hence, its recall can be considered the upper bound for this value, which is inherited somehow by OntoTagger.

Regarding the values of the indicators in the second group, they are referred to as *Statistics from n° words with category matching between tools* in Figure 27. As can be seen in the aforementioned figure, OntoTagger outperforms DataLexica in the 27.32% of the cases (the respective value of this comparative indicator is shown in blue in the bar chart), and FDG in the 12.34% of the cases (the respective value of this comparative indicator is shown in maroon in the bar chart). However, once again, the third value of this comparative indicator shows that OntoTagger and UMurcia are equivalently accurate, which results from the same reasons described above.

So, according to the results presented in the current subsection, it can be considered that OntoTagger outperforms most of the other tools integrated into its configuration. Hence, also in this case, the aforementioned underlying hypothesis is supported by the empirical results, which also allow for a positive evaluation of the corresponding objectives of OntoTag.

6.1.2. STATISTICAL ANALYSIS OF THE LEMMA COMBINATION RESULTS

In this subsection, it is presented the particular statistical indicators of the lemma combination sub-phase. They give an idea of the precision of the lemma tag associated, by each of the tools considered (namely UMurcia, FDG, DataLexica and OntoTagger), to the different morphosyntactic units of the input text. The values of the particular indicators associated to this combination sub-phase are shown in Figure 28.

The bar chart included in Figure 28 presents the statistical indicators associated to the lemma tags assigned to the items in the input text. The results of each group of statistical indicators, namely *Correct*, *Wrong* and *Null*, are calculated (respectively) as the arithmetic mean of the items correctly, incorrectly and not annotated by the corresponding tool. They have been grouped together, in order to show more clearly the behaviour of each tool, *i.e.*, OntoTagger (in dark blue), DataLexica (in maroon), FDG (in beige), and UMurcia (in light blue), with respect to the other ones.

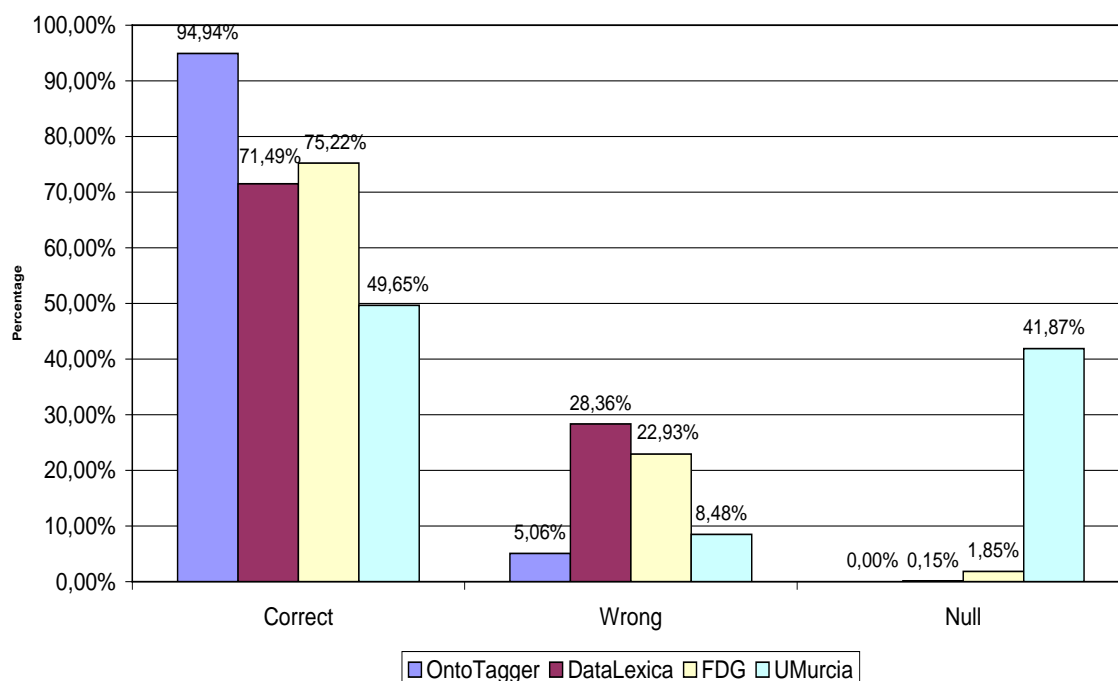


Figure 28: Comparative statistics associated to the precision of lemma tags

As can be seen in this figure, the value of the *Correct* indicator for OntoTagger (94.94%) highly outperform those of DataLexica (71.49%), DataLexica (75.22%) and UMurcia (49.65%).

As for the values of the *Wrong* indicator, it can be observed that the one corresponding to OntoTagger (5.06%) clearly outperforms, once again, those of the other tools. Whereas a high value of the previous indicator entails a higher number of correctly annotated tags and, hence, points out a good performance of the tool under consideration, a high value of the present indicator points out a bad performance, since it entails a higher number of wrongly annotated items. Accordingly, OntoTagger results outperform five times those of DataLexica (28.36%), and four times those of FDG (22.93%). They also surpass the results of UMurcia (8.48%).

As for the values of the *Null* indicator, it can be observed that the one corresponding to OntoTagger (0.00%) outperforms, once more, those of the other tools (also in this case, a high value of the indicator points out a bad performance). Accordingly, OntoTagger results outperform slightly those of DataLexica (0.15%), and FDG (1.85%), and highly those of UMurcia (41.87%).

A remark must be made about the results of UMurcia shown above, which should not be misunderstood. The problem of UMurcia annotations is that they lacked those associated to the items included as HTML links in the web pages of the ODECORPUS-Entertainment, for example. Hence, the corresponding items were assigned automatically the value “null” when combining the results of the different tools incorporated into OntoTagger.

In any case, according to the results presented in the current subsection, especially those concerning the first and the third indicators, it can be considered that OntoTagger outperforms (highly) the rest of the tools. Hence, once more, they support the aforementioned underlying hypothesis, and also allow for a positive evaluation of the corresponding objectives of OntoTag.

6.1.3. STATISTICAL ANALYSIS OF THE MORPHOLOGICAL ATTRIBUTE COMBINATION RESULTS

In this subsection, it is presented the generic statistical indicators of the morphological attribute combination sub-phase. They give an idea of the precision of the morphological attribute tag assigned, by each of the tools considered (namely UMurcia, FDG, DataLexica and OntoTagger), to the different morphosyntactic units of the input text. The values of the particular indicators associated to this combination sub-phase are shown in Figure 29.

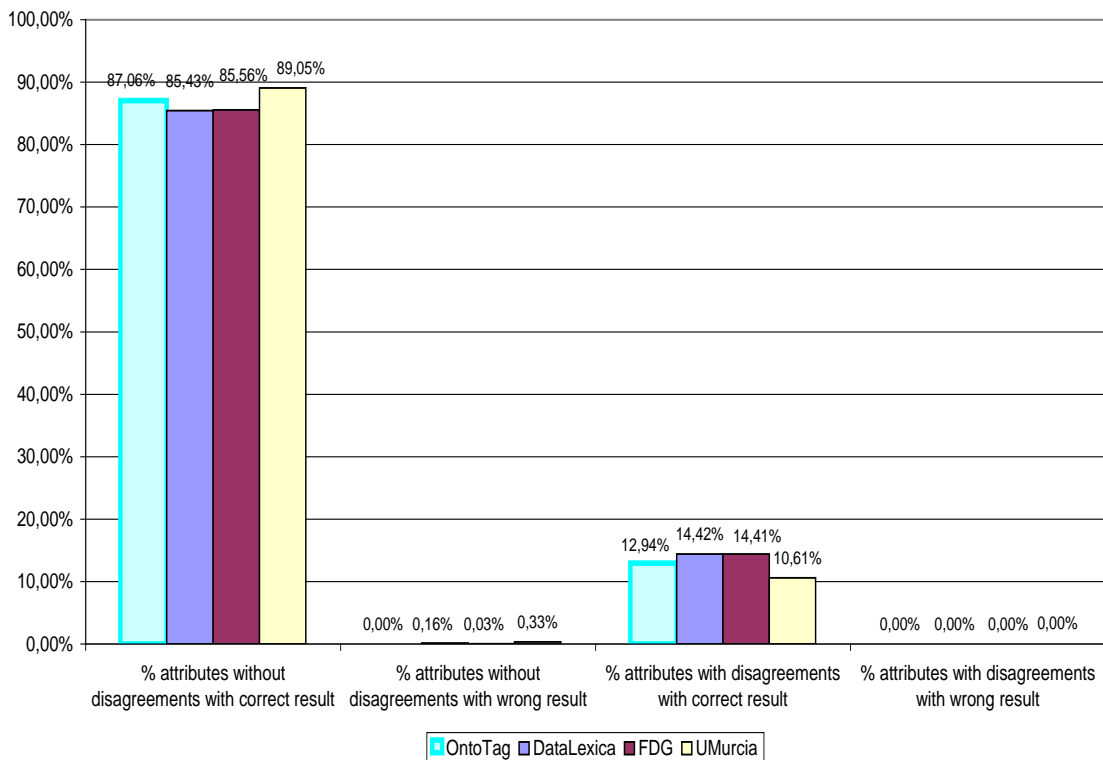


Figure 29: Comparative statistics associated to the recall of the values of morphological attributes

The bar chart included in Figure 29 presents the statistical indicators associated to the morphological attribute tags assigned to the items in the input text (loosely referred to as *(%)attributes*). The results of each statistical indicator, namely *(%)attributes without disagreements with correct result*, *(%)attributes without disagreements with wrong result*, *(%)attributes with disagreements with correct result*, and *(%)attributes with disagreements with wrong result*, have been calculated as explained in the introduction of Section 6.1.1.1. Once again, they have been grouped together in order

to show more clearly the behaviour of each tool, namely OntoTagger (in light blue), DataLexica (in dark blue), FDG (in maroon), and UMurcia (associated to LACELL's tagger, in beige), with respect to the other ones.

The first group of indicators in Figure 29, referred to as *(%)attributes without disagreements with correct result*, shows the average number of morphological attributes in the input text that were annotated correctly by each tool. This entails that the corresponding tag coincided with the one assigned to the same item in the gold standard (*i.e.*, the corpus sample annotated and manually corrected beforehand). As commented above, this statistical indicator takes into account that all the tools assigned the same morphological attribute tag to the morphological attribute in question (that is, the tags were combined *without disagreements*).

As can be seen in the aforementioned figure, the value of this particular indicator for OntoTagger (87.06%) slightly outperforms those of DataLexica (85.43%) and of FDG (85.56%), but not those of UMurcia (89.05%). However, due to the extremely low difference in their values, this statistical indicator cannot be considered significant.

The second group of indicators in Figure 29, referred to as *(%)attributes without disagreements with wrong result*, shows the average number of morphological attributes in the input text that were annotated incorrectly by each tool. This entails that the corresponding tag did not coincide with the one assigned to the same item in the gold standard. As commented above, this statistical indicator takes into account that all the tools assigned the same morphological attribute tag to the morphological attribute in question (that is, the tags were combined *without disagreements*).

As shown in the figure, the value of this particular indicator for OntoTagger (0.00%) slightly outperforms those of the other tools. As commented previously, whereas a high value of the previous indicator entails a higher number of correctly annotated tags and, hence, points out a good performance of the tool under consideration, a high value of the present indicator points out a bad performance, since it entails a higher number of wrongly annotated items. Accordingly, OntoTagger outperforms DataLexica (0.16%), UMurcia (0.03%) and FDG (0.33%). All of these values are so low and similar, that neither this second statistical indicator would be considered significant, yielded it not a 0.00% of tagging errors for OntoTagger. Actually, this value means that it completely eradicates this type of errors, whereas the rest of the tools don't.

The third group of indicators in Figure 29, referred to as *(%)attributes with disagreements with correct result*, shows the average number of morphological attributes in the input text that were annotated correctly by each tool. However, in this case, this indicator summarises the results for the cases in which not all the tools assigned the same morphological attribute tag to the morphological attribute in question (that is, the tags were combined *with disagreements*).

As can be seen also in the figure, the values of this particular indicator for DataLexica (14.42%) and of FDG (14.41%), most similar, slightly outperform those of OntoTagger (12.94%) and UMurcia (10.91%). This clearly inverts the trend pointed out by the first group of statistical indicators for morphological attribute tags. Most opportunely, the first and the third groups of statistical indicators can be associated by virtue of them both being related to correct results. Accordingly, from this perspective, what these two groups of indicators show globally is that all the tools considered perform similarly (well) in the annotation of morphological attributes, and that (bad) the results in one of the groups are compensated by the (good) ones in the other.

Finally, the fourth group of indicators in Figure 29, referred to as *(%)attributes with disagreements with wrong result*, shows the average number of morphological attributes in the input text that were annotated incorrectly by each tool. Also in this case, this indicator summarises the results for the cases in which not all the tools assigned the same morphological attribute tag to the morphological attribute in question (that is, the tags were combined *with disagreements*).

As can be seen in the figure mentioned above, the values of this particular indicator for each and every tool coincide and equal 0.00%, which, considered alone, would make this indicator rather insignificant. However, as with the first and the third groups of statistical indicators, also the second and the fourth groups can be associated but, in this case, by virtue of them both being related to wrong results. Accordingly, from this perspective, these two groups of indicators yield a most significant result with respect to the evaluation of OntoTagger (and, hence, of OntoTag): whereas OntoTagger completely eradicates all types of errors in the combined annotation of morphological attributes, the other tools (separately) don't.

So, according to the results presented in the current subsection, especially those concerning the second and the fourth indicators, it can be considered that OntoTagger outperforms the rest of the tools. Hence, also in this case, the aforementioned underlying hypothesis is supported by the empirical results, which also allow for a positive evaluation of the corresponding objectives of OntoTag.

To conclude the present section, dealing with the evaluation of the whole combination sub-phase of OntoTag by means of the results yielded by OntoTagger, **it has been shown that OntoTagger clearly outperformed two of the tools integrated into its configuration, namely DataLexica and FDG** in all the combination sub-phases in which they overlapped. As far as the remaining tool is concerned, *i.e.*, **LACELL's tagger (aka UMurcia)**, **it was also outperformed by OntoTagger in the annotation of morphosyntactic categories and lemmas**, but only behaved slightly better in the sub-phase of morphological attribute combination.

However, **these results** suffice for an overall positive evaluation of the combination sub-phase of OntoTagger, which entails a minimal validation of the objectives of the model OntoTag, and **supports**

the corresponding underlying hypothesis for the whole phase as well. **That is**, in few words, **that the combination of the annotations of several tools operating at the same level (usually) outperforms the results obtained when annotating with the tools separately.** *As an immediate result, this implies that this type of combination architecture configurations can be applied in order to improve significantly the accuracy of linguistic annotations.*

Some additional results, which reinforce the previously mentioned conclusions, were generated from some particular type of OntoTagger annotations lying at the `Semantic Level`. This type of semantic annotations is performed by the named entity recognition and classification subsystem of OntoTagger, presented in Section 5.4. Its associated results are presented and discussed in the following subsection.

6.2. EVALUATION OF THE NAMED ENTITY RECOGNITION AND CLASSIFICATION SUBSYSTEM

As shown in a previous chapter, in order to evaluate the suitability of the model OntoTag at the `Semantic Level`, a particular module was expressly developed to produce, integrate and combine different sources and layers of semantic tagging within OntoTagger's results, using the combined and improved morphosyntactic and syntactic annotations previously obtained by means of this configuration.

This semantic module of OntoTagger included, amongst others, an expressly developed subsystem for the recognition, classification and labelling of named entities, according to the MUC and ACE tagsets (further specified by means of the CNEO domain ontology, introduced above). This subsystem implemented some heuristic rules, based on linguistic clues, provided by the linguist team of the OEG. They detailed the linguistic patterns that hinted the existence of a named entity in the documents of the domain in question (entertainment).

As in the combination sub-phase of OntoTagger, first, a small sample²¹⁸ (also consisting of the first ten pages or, equivalently, of around 5000 morphosyntactic units) of ODECORPUS-Entertainment was morphologically and morphosyntactically annotated, using OntoTagger. Second, the named entities of this small corpus sample were manually annotated, according to the MUC-7 and ACE recommendations, and a new XML document containing the results of this named entity manual annotation was generated (referred to as the gold standard henceforth). Third, the annotations obtained in the first phase were used to apply the heuristic rules aforementioned and perform the corresponding

²¹⁸ Also the scope of these experiments had to be reduced to a minimum, due to time and funding constraints.

named entity recognition, classification and labelling. Fourth, the results of this named entity (automatic) labelling were compared with the ones included in the gold standard.

In this subsection, we present the (statistical) results obtained from the evaluation of this particular subsystem of the semantic module of OntoTagger. The statistical measures applied are the usual ones, borrowed from the Information Retrieval field (*i.e.*, precision, recall and E-measure), but particularised to a named entity recognition and classification scenario.

Hence, the statistical measures of the first group applied to the results of the present task were:

- **Precision** (aka P , shown in Table 146, and conveniently summarised in Figure 30 as well): Details the ratio of the number of named entities retrieved with respect to the total number of named entities annotated in the gold standard.
- **Recall** (aka R , also shown in Table 146, and conveniently summarised in Figure 31): Details the ratio obtained as the result of dividing the number of actual named entities retrieved (or, equivalently, the number of items retrieved correctly) by the total number of items retrieved (either correctly or incorrectly).
- **E-Measure** (aka E): Results from the combination of the two previous ones, by means of the accompanying formula, in which:

- P = Precision
 - R = Recall
 - b = a weight(ing) value that measures the relative relevance given to P and R (when $b = 0.5$, both indicators are equally relevant).
- $$E = 1 - \frac{b^2 PR + PR}{b^2 P + R}$$

The resulting values of P , R and E , for the ten corpus files annotated are included in Table 146. They were calculated assigning b the value 0.5. **As can be observed in the aforementioned table and figures, the results are extraordinary good for such a task.**

These results can be explained on the basis of (i) the high accuracy of the annotations provided by OntoTagger at the lower levels (mainly at the morphosyntactic level); and (ii) the heuristic rules that implement the linguistic patterns supplied by the OEG's linguist team towards this end. However, these results should be conveniently qualified, since they might be too domain- and/or language-dependent. It should be further experimented how similar rules work in a different domain or a different language, such as French, English, or German.

Table 146: Detailed precision and recall calculation for the ODECORpus-Entertainment testing sample

FILE	RECALL	PRECISION	E-MEASURE
1	$(64 / 71) * 100 = 90,14\%$	$(64 / 64) * 100 = 100\%$	0,0805
2	$(51 / 57) * 100 = 89,47\%$	$(51 / 51) * 100 = 100\%$	0,0860
3	$(39 / 40) * 100 = 97,50\%$	$(38 / 39) * 100 = 97,44\%$	0,0251
4	$(32 / 37) * 100 = 86,49\%$	$(30 / 32) * 100 = 93,75\%$	0,1215
5	$(38 / 41) * 100 = 92,68\%$	$(38 / 38) * 100 = 100\%$	0,0594
6	$(64 / 70) * 100 = 91,43\%$	$(62 / 64) * 100 = 96,88\%$	0,0753
7	$(47 / 63) * 100 = 76,40\%$	$(47 / 47) * 100 = 100\%$	0,2140
8	$(37 / 44) * 100 = 84,09\%$	$(37 / 37) * 100 = 100\%$	0,1315
9	$(44 / 49) * 100 = 89,80\%$	$(44 / 44) * 100 = 100\%$	0,0833
10	$(80 / 87) * 100 = 91,95\%$	$(80 / 80) * 100 = 100\%$	0,0654
Total	$(496 / 559) * 100 = 88,73\%$	$(491 / 496) * 100 = 98,99\%$	0,0939

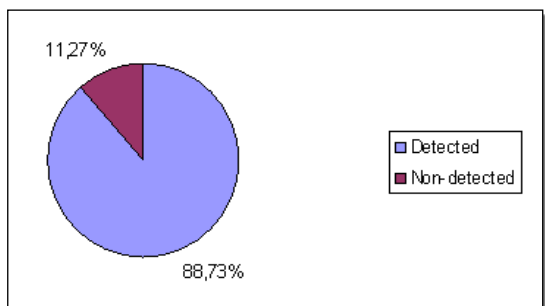


Figure 30: Overall recall of OntoTagger on named entity recognition and classification (or annotation)

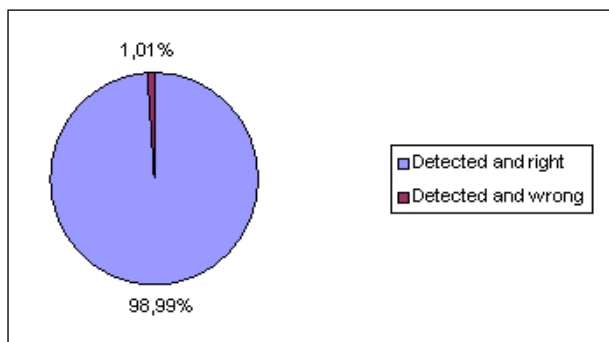


Figure 31: Overall precision of OntoTagger on named entity recognition and classification (or annotation)

In any case, as commented previously, the named entities recognised, sub-classified and labelled by means of this subsystem were used to populate the CNEO. The same statistics that apply for the labelling (or annotation) task apply also for the task of ontology population. Hence, the results of the present application of Human Language Technologies to Ontology Population (and, accordingly, to Ontological Engineering) seem very promising and encouraging in order for these two areas to collaborate and complement each other.

7. CONCLUSIONS

After analysing and evaluating the main results yielded by OntoTagger (the first implementation of the OntoTag model) this chapter summarises the main conclusions derived from the development of the present work. They are related to the development of both the model and its implementation. Firstly, the main contributions of this work are shown in Section 7.1. Secondly, Section 7.2 discusses its other practical outcomes, including a set of recommendations, best practices and lessons learned for the standardisation, interoperability and merge of linguistic tools and annotations (Subsection 7.2.1). Thirdly, Section 7.3 presents some other theoretical outcomes of the present thesis. And, finally, Subsection 7.4 surveys the assessment of the hypotheses of this thesis, which were introduced in Subsection 2.4.2 (page 27).

7.1. MAIN OUTCOMES

1. As commented in the previous chapters, the global aim of the present work was to develop a **hybrid (semantic) annotation model (OntoTag), that is, linguistically motivated and ontology-based, suitable for the Semantic Web**. The OntoTag hybrid model is, thus, its first main outcome as well.

1.1. **This hybrid model for annotation consists of**

1.1.1. **An abstract scheme that helps merge the morphosyntactic, syntactic and semantic annotations of written texts;**

1.1.2. **An abstract architecture that allows several different linguistic annotation tools and their corresponding annotations to interoperate.**

1.2. OntoTag benefits from the scientific and technological advances in both Artificial Intelligence (and/or the Semantic Web) and Corpus Linguistics (and, more concretely, of Linguistic Annotation), and minimises the impact of the lacks and research gaps of both of them. Accordingly,

1.2.1. As in the Semantic Web, **OntoTag's conformant annotations are ontology-based annotations**, that is,

- They make meaning explicit by means of ontologies (*i.e.*, its annotation labels are ontology terms);
- They use ontology terms within <Subject, Predication, Object> triples, in order to link them and constitute standardised and formal annotations;
- Consequently, they are machine-readable (*i.e.*, computer understandable);

1.2.2. Besides, **from a linguistic annotation point of view, these annotations**

1.2.2.1. **Can be considered either**

- **a kind of semantic field annotation** (if an upper-level ontology is being used);
- **a kind of sense tagging** (if a domain ontology is being used).

1.1.1.2. **Can be applied not only to the semantic level**, as they have been applied within the Semantic Web, **but also to the rest of linguistic levels**.

1.1.1.3. **Comply both with the EAGLES (1996a; 1996b) recommendations and with the most recent ISO standards drafts (e.g., ISO/LAF (2009)), by means of their <Linguistic Unit²¹⁹, Linguistic Attribute, Linguistic Value> triple structure, which is recommended by these works²²⁰.**

2. Obviously, (i) to be ontology-based, such a hybrid model required an ontological formalization of the elements being annotated; and (ii) to be linguistically-motivated, the elements being annotated had to be of a linguistic nature. Hence, **the elements involved in linguistic annotation were formalised in a set (or network) of ontologies. This is the second main outcome of this work: OntoTag's linguistic ontologies.**

2.1. OntoTag's network of ontologies consists of

2.1.1. *The Linguistic Unit Ontology (LUO)*, which includes a mostly hierarchical formalisation of the different types of linguistic elements (*i.e.*, units) that can be identified in a written text;

2.1.2. *The Linguistic Attribute Ontology (LAO)*, which includes also a mostly hierarchical formalisation of the different types of features that characterise the linguistic units included in the LUO;

2.1.3. *The Linguistic Value Ontology (LVO)*, which includes the corresponding formalisation of the different values that the attributes in the LAO can take;

2.1.4. *The OIO (OntoTag's Integration Ontology)*, which

- Includes the knowledge required to link, combine and unite the knowledge represented in the LUO, the LAO and the LVO;
- Can be viewed as a knowledge representation ontology that describes the most elementary vocabulary used in the area of annotation.

2.2. **OntoTag's ontologies incorporate the knowledge included in the different standards and recommendations for linguistic annotation released so far**, such as those developed within the EAGLES and the SIMPLE European projects or by the ISO/TC 37 committee:

2.2.1. As far as morphosyntactic annotations are concerned, OntoTag's ontologies formalise the terms in *the EAGLES (1996a) recommendations* and their corresponding terms

²¹⁹ Or else Linguistic Category.

²²⁰ This triple structure was included in EAGLES (1996a) (i) complies with the W3C-compliant <Subject, Predication, Object> triple structure; and (ii) has been inherited and standardised within the most recent ISO standards (ISO/LAF, 2009; ISO/MAF, 2008).

within *the ISO Morphosyntactic Annotation Framework (ISO/MAF, 2008) standard*;

2.2.2. As for syntactic annotations, OntoTag's ontologies incorporate the terms in *the EAGLES (1996b) recommendations* and their corresponding terms within *the ISO Syntactic Annotation Framework (ISO/SynAF, 2010) standard draft*;

2.2.3. Regarding semantic annotations, OntoTag's ontologies generalise and extend the recommendations in EAGLES (1996a; 1996b) and incorporate *the terms in SIMPLE (2000)*²²¹;

2.2.4. The terms coming from all these recommendations and standards were supplemented by those within *the ISO Data Category Registry (ISO/DCR, 2008)*²²² and also of *the ISO Linguistic Annotation Framework (ISO/LAF, 2009) standard draft* when developing OntoTag's ontologies.

7.2. OTHER PRACTICAL OUTCOMES

1. The first practical outcome concerning standardisation is **the extension of the EAGLES (1996a; 1996b) recommendations to detail the concepts and the layers involved in the morphosyntactic, syntactic and the semantic annotation of texts**. It *resulted from the development of OntoTag's annotation scheme*.
2. In addition, *the whole set of terms included into OntoTag's ontologies can be viewed as a general-purpose, structured and standardised set of linguistic data categories (LDCs)* for the linguistic annotation of web documents at the morphosyntactic, the syntactic and the semantic levels.
 - 2.1. **This general-purpose set of LDCs is suitable not only for the Semantic Web** (the context in which the present research has been carried out), **but also for other areas and projects**, such as **Corpus Linguistics, and more concretely, for the ISO/DCR international standard draft**.
 - 2.2. **Furthermore, their hierarchical arrangement into an ontology allows also for varying degrees of granularity in annotations** (Aguado *et al.*, 2002; Ide, Romary & de la Clergerie, 2003). This issue was already present in EAGLES (1996a, 1996b); however, it has been neglected somehow in most ISO standard proposals for linguistic annotation (such as ISO/MAF or ISO/SynAF).

²²¹ No stable standard draft has been released for semantic annotation by ISO/TC 37 yet.

²²² From the ISO/TC 37/SC3 subcommittee. See also <http://www.isocat.org/>.

7.2.1. RECOMMENDATIONS, BEST PRACTICES AND LESSONS LEARNED FOR ANNOTATION STANDARDISATION, INTEROPERATION AND MERGE

1. **OntoTag's abstract architecture for the hybrid annotation of texts can be viewed as a methodology and a best practice for (a) reusing, standardising and merging linguistic annotations; and (b) integrating different linguistic annotation tools and making them interoperate. Thus,**

1.1. *The following processes should be implemented in any tool aiming at merging linguistic annotations:*

1.1.1. The *standardisation processes* (i) that allow for complying with the standardisation requirements of the abstract scheme of OntoTag; and (ii) that facilitate the decanting and the combination of the results of those tools that share some level of annotation.

1.1.2. The *decanting processes* that allow for the classification and separation of the results of those tools performing their annotations at several different levels.

1.1.3. The *merging processes* that allow for the integration of the results and the interoperation of the tools, that is,

1.1.3.1. The *combination processes* for merging the annotations of a given level.

1.1.3.2. The *integration processes* for merging the annotations of different levels.

2. Some experiments carried out with OntoTagger showed that

2.1. **Standardisation processes should be run before decanting processes.** This is the most efficient way to arrange these two phases.

2.1.1. For instance, whereas some tools might compress somehow their morphosyntactic annotations, tagging more than one token as a single morphosyntactic unit, some others might not. Decanting before standardising entailed multiple unnecessary checks and corrections of the corresponding segmentations, which should be avoided.

2.2. **The decanting processes should generate a set of up to four different outputs and/or documents²²³**, in order to separate the morphosyntactic, syntactic and semantic input annotations, according to their level and layer:

2.2.1. One document containing both the lemma and the morphosyntactic category annotations (**L+POS**);

2.2.2. One consisting of the morphosyntactic category and the morphological annotations (**POS+M**),

2.2.3. One more for the syntactic counterpart of the annotations (**Syn**);

²²³ Depending on the level(s) annotated by each tool: the more levels annotated, the more documents generated.

2.2.4. Another one for the semantic annotations (**Sem**).

This way,

- i) The remaining phases are not complicated;
- ii) The comparison, evaluation and mutual supplementation of the results offered at the same level by different tools is simplified;
- iii) The decanted results can be easily integrated, after they have been compared and combined.

2.3. Concerning the *combination processes*, we recommend the following²²⁴ ordering for the combination and integration of the different annotations:

2.3.1. *Syntactic combination (A)*: The process of syntactic combination should be split and, first of all, any discontinuity or incoherence in the numbering of the tokens across the different documents should be discovered and solved. In this first subprocess,

- 2.3.1.1. All the different decanted documents being combined at the syntactic level shall be scanned in parallel for morphosyntactic gaps, which should be conveniently filled.
- 2.3.1.2. At the same time, the tokens marking up the ends of sentence or paragraph in the annotations of each tool shall be identified and unified.
- 2.3.1.3. This subprocess ends when a final and unified numbering has been achieved and all changes have been propagated to the rest of the decanted documents (at every level).

2.3.2. *L+POS combination*:

- 2.3.2.1. Morphosyntactic categories (POSs) should be combined immediately after syntactic combination (A), and the results of POS combination shall be propagated to the rest of the levels and layers (i.e., the morphological feature layer, as well as the syntactic and the semantic levels).
- 2.3.2.2. Lemma combination should be carried out after POS combination has been accomplished.

2.3.3. *POS+M combination*: Morphological attributes and their values should be combined after L+POS combination.

- Exceptionally, this process could be executed in parallel with the following one.

2.3.4. *Syntactic combination (B)*: The second syntactic combination subprocess should be run after the morphosyntactic combination. This subprocess should

- 2.3.4.1. Combine together the annotations of the same syntactic layer coming from different tools.

²²⁴ Empirically-determined.

2.3.4.2. Unite and interrelate afterwards all the different syntactic annotation layers in a one and only document containing the overall conjoined syntactic annotations for the input document.

2.3.5. **Semantic combination:** Finally, the semantic annotations of the input document coming from different tools should be combined as well, layer by layer (possibly in parallel).

2.3.5.1. In OntoTagger, this process required all the lower levels to be annotated before being executed.

2.3.5.2. Besides, in most cases,

i) sense tagging is built on top of a POS tagging.

ii) a certain degree of parsing (i.e., syntactic annotation) seems indispensable for deeper and broader semantic analyses and annotations.

2.3.5.3. As a consequence, *there is little chance that this process be executed in parallel with any of the others.*

2.3.5.4. Thus, **semantic combination should be placed in the last position of the integration phase.**

2.4. **Or else, in any case,**

2.4.1. **Morphosyntactic category combination shall be performed before lemma combination.**

2.4.2. **Morphosyntactic category combination shall be performed before morphological feature combination.**

2.4.3. **Syntactic unification and segmentation into paragraphs, sentences and tokens shall be done before executing any other combination process.** This allows achieving an accurate and consistent numbering of tokens before proceeding with the integration phase.

3. Besides, **OntoTag's abstract architecture includes a tool- and language-independent algorithm for specifying and implementing *combination processes*.** This method consists of five steps, namely,

Step 1. Identifying the weaknesses and the strengths of each linguistic tool to be incorporated into the configuration. The final aim of this thorough analysis is to obtain a list of annotation failures (characterised as annotation errors, gaps and underspecifications) per tool.

Step 2. Characterising statistically each of these failures with respect to the linguistic contexts in which it occurs.

Step 3. Inferring an abstract model (that is, a specification) that explains the behaviour of the tools analysed. This abstract model should include some patterns that detail the typical

failures found for each tool (or, at least, the most common ones), as well as the corresponding hints (or heuristics) for their correction.

Step 4. Selecting the particular and most suitable combination methods (production rules –with or without probabilistic triggering factors–, Bayesian networks, etc.) to implement the aforementioned model.

Step 5. Implementing the resulting model, together with its combination methods. We recommend following an incremental prototyping cycle for this. This incremental prototyping development should implement, in each phase, a gradually growing set of failure-correcting hints. This entails determining, after each phase is completed, if the last set of hints implemented introduces new failures (and, hence, it should be removed, substituted or refined) or else, if it removes successfully the corresponding failure (and, hence, it should be consolidated into the module).

7.3. OTHER THEORETICAL OUTCOMES

Several other theoretical outcomes have been obtained by means of this work. They are presented in this subsection according to their type: on the one hand, standardisation-related theoretical outcomes are included in Subsection 7.3.1; on the other hand, annotation interoperation and merge theoretical outcomes are discussed in Subsection 7.3.2.

7.3.1. CONCERNING STANDARDISATION

1. As discussed in the State of the Art (Section 3.2.1.3.2.2, page 49), Schmidt’s criteria are the best reference publicly available so far for developing standardised semantic annotation tagsets. **This work shows that the use of ontologies as a basis for a semantic annotation scheme accomplishes Schmidt’s criteria.** Table 147 (see next page) shows how each Schmidt’s criterion has been fulfilled by *OntoTag*’s ontologies, which constitute the tagset associated to *OntoTag*’s annotation scheme.
2. Furthermore, **as *OntoTag*’s ontologies have been developed following the existing standards, guidelines and recommendations for annotation, annotating with reference to them produces a result that uses a standardised type of tagset.** The tagsets and annotations of each and every tool are mapped in the phase of standardisation onto the terms of *OntoTag*’s ontologies and, hence, all phenomena being tagged share a common and standardised vocabulary for their description. In addition, this common and standardised vocabulary can be considered formal and fully semantic, from a computational point of view, since it is referred to ontologies. Accordingly, it can be concluded that *OntoTag*’s ontologies (can) play a crucial role in the standardisation of linguistic annotations.

Table 147: Evaluation of OntoTag's scheme against Schmidt's criteria

SCHMIDT'S CRITERION	FULFILLMENT WITHIN ONTOTAG
1. Making sense in linguistic terms	OntoTag's ontologies are a formal and computational representation of linguistic terms.
2. Being able to account exhaustively for the vocabulary in the corpus	OntoTag's ontologies can easily grow horizontally (in breadth) and vertically (in depth) to cover all the vocabulary in the corpus.
3. Being sufficiently flexible to allow for those emendations that might be necessary.	The classes in OntoTag's ontologies can be specialised according to new criteria and sub-classifications.
4. Operating at an appropriate level of granularity.	i) OntoTag's ontologies can easily grow vertically (in depth), if a finer-grained tagset is required. ii) Conversely, finer-grained terms can be pruned if a coarser-grained tagset is required.
5. Possessing a hierarchical structure.	Fulfilled by the mostly hierarchical structure of OntoTag's ontologies.
6. Conforming to a standard, if one exists.	Most of the linguistic terms in OntoTag's ontologies are or will be included in the ISO/DCR international standard draft and are, thus, standard-conformant.

3. **A final spin-off of the standardisation phase is that it enabled the model to handle the annotations performed by any linguistic tool, irrespective of the level(s) they annotated at and the schemas (or the tagsets) employed.** Indeed, *after the document being annotated was processed in the phase of standardisation, the annotations of the same phenomenon coming from the different tools complied with the same schema and used the same tagset. They were, thus, comparable.*

3.1. **A major drawback, though, is that it required a previous study of the output schemas and the tagsets of each of the tools assembled into the architecture. Indeed, their interpretation and mapping onto the standardised scheme and the ontology-based tagset of OntoTag could not be determined automatically *a priori*.** Consequently, an *ad-hoc*, tool-dependent standardising wrapper had to be implemented for each linguistic annotation tool assembled into the architecture.

7.3.2. CONCERNING ANNOTATION INTEROPERATION AND MERGE

1. **The uniform and structured view of linguistic annotation provided by (i) OntoTag's annotation scheme and (ii) OntoTag's ontologies and the linguistic data categories that they formalise, helped immensely in**

1.1. **The decanting of linguistic annotations.**

1.2. **The merge and interoperation of different annotations, that is,**

- 1.2.1. **The integration (and interoperation) of morphosyntactic, syntactic and semantic annotations into a unique (but multileveled) annotation of the input text.**
- 1.2.2. **The combination of the annotations pertaining to the same level into a unique annotation per level.** In effect, the annotations of each level were collected according to their associated top-level concept in the LUO, which detailed the linguistic level to which each annotation belonged.
2. In addition, we propose **several mathematically-founded notations and formalisms for the formalisation of combination subprocesses.** These notations and formalisms are **based on a number of well-known mathematical concepts, such as *multiset*, *partial order*, or *lattice* and their accompanying algebraic theories.** Accordingly, they can also be implemented and processed by means of several libraries and algorithms already developed for this purpose.
3. Finally, the integration in OntoTag and OntoTagger of semantic annotations with the remaining ones, lying at lower levels, shows that
 - 3.1. **Linguistic annotations can be joined and linked together (that is, interoperate) at a larger scale.**
 - 3.2. **The corresponding annotation processes and tools can interoperate as well, in order to obtain this type of joined, linked, multilayered and multilevel annotations.**
 - 3.3. Moreover, **OntoTag's combination configurations can be applied to improve significantly the accuracy and/or the robustness of linguistic tools and/or annotations.**
 - 3.3.1. However, the relevance of the different technological (stochastic *vs.* rule-based, for example) or theoretical (dependency *vs.* HPS-grammar-based, for instance) approaches followed in the development of the tools incorporated into OntoTagger could not be evaluated by means of the experiments carried out and should be explored in further research.

7.4. ONTOTAG'S HYPOTHESES ASSESSMENT

In this section, we review and survey our conclusions concerning OntoTag's hypotheses, introduced in Subsection 2.4.2 (page 27), but re-written here for the sake of clarity. The assessment of OntoTag's hypotheses has been included in Table 148 (page 334). Each row of this table comprises (a) the identifier for the hypothesis; (b) the hypothesis itself; (c) the way in which the hypothesis was evaluated and/or assessed; and (d) whether it was confirmed or rejected by this evaluation and/or assessment. As shown in the table, only one of the hypotheses (H.6) was rejected; the other five could be confirmed.

Table 148: Assessment of OntoTag’s hypothesis

IDENT.	HYPOTHESIS		ASSESSED BY MEANS OF	ASSESSMENT RESULT
H.1	The annotations of different levels (or layers) can be integrated into a sort of overall, comprehensive, multilayer and multilevel annotation, in order for their elements to complement and refer to each other.		The development of: <ul style="list-style-type: none"> • OntoTag’s annotation scheme, • OntoTag’s annotation architecture, • OntoTagger’s annotation (XML, RDF and OWL) schemas, • OntoTagger’s configuration. 	Confirmed.
H.2	Tool-dependent annotations can be mapped onto a sort of tool-independent annotations and, thus, be standardised.		The standardisation phase incorporated into OntoTagger for the annotations yielded by the tools.	Confirmed.
H.3	Standardisation should ease	H.3.1: The interoperation of linguistic tools. H.3.2: The comparison, combination (at the same level and layer) and integration (at different levels or layers) of annotations.	The development of OntoTagger’s ontology-based configuration: <ul style="list-style-type: none"> • Interoperation, comparison, combination and integration of the annotations of three different linguistic tools (FDG, Bitext’s DataLexica and UMurcia POS tagger); • Integration of EuroWordNet-based, domain-ontology-based and named entity annotations at the semantic level. • Integration of morphosyntactic, syntactic and semantic annotations. 	Confirmed.
H.4	Ontologies and Semantic Web technologies (can) play a crucial role in the standardisation of linguistic annotations, by providing consensual vocabularies and standardised formats for annotation (e.g., RDF triples).		The development of OntoTagger’s RDF-triple-based annotation schemas.	Confirmed.

IDENT.	HYPOTHESIS	ASSESSED BY MEANS OF	ASSESSMENT RESULT
H.5	The rate of errors introduced by a linguistic tool at a given level when annotating can be reduced automatically by contrasting and combining its results with the ones coming from other tools operating at the same level. However, these other tools might be built following a different technological (stochastic <i>vs.</i> rule-based, for example) or theoretical (dependency <i>vs.</i> HPS-grammar-based, for instance) approach.	The results yielded by the evaluation of OntoTagger.	Confirmed ²²⁵ .
H.6	Each linguistic level can be managed and annotated independently.	OntoTagger's experiments and the dependencies observed among the morphosyntactic annotations and between them and the syntactic annotations ²²⁶ .	Rejected.

²²⁵ OntoTagger clearly outperformed two of the tools incorporated into its configuration, namely DataLexica and FDG in all the combination sub-phases (dealing, respectively, with morphosyntactic categories, lemmas and morphological attributes). As far as the remaining tool is concerned, *i.e.*, LACELL's tagger (aka UMurcia), it was also outperformed by OntoTagger in the annotation of morphosyntactic categories and lemmas, though it behaved slightly better in the sub-phase of morphological attribute combination.

²²⁶ When OntoTag's ontologies were developed, we observed that several linguistic units stand on an interface between levels, belonging thereby to both of them (such as morphosyntactic units, which belong to both the Morphological Level and the Syntactic Level). Therefore, the annotations of these levels overlap and cannot be handled independently when merged into a unique multileveled annotation.

8. FURTHER WORK

When developing OntoTag and OntoTagger, we have identified a huge amount of work that had to be left for further research, since it fell out of the scope of this thesis and/or had to wait for later (for example, the final linking to ISO standards, which has to wait until they are eventually released).

Maybe **the first task to face** after the present research **should be to complete the semantic level of OntoTag's ontologies (in particular, the LUO, the LAO and the LVO) and to extend them to other levels of linguistic description (such as Morphology, Discourse or Pragmatics)**. In fact, these extensions have already been prospected before writing the present dissertation, in order to provide these ontologies with an evaluation of their extensibility. Even though the preliminary results were very promising, this task had to be abandoned for the sake of time. Another interesting extension would be to **fully develop the Linguistic Level Ontology (LLO)**, by formalising the different layers of annotation already identified in the present work, and also many subdivisions that have been posed by the research carried out hitherto.

This first task entails a continued effort to keep OntoTag's ontologies linked to the linguistic data categories (LDCs) included in the ISO Data Category Registry (ISO/DCR). The ISO/DCR has been searched continuously for the LDCs relevant to this work. Following this ISO standard proposal, the corresponding Data Category Selections (DCSs) have been extracted and mapped onto OntoTag's ontologies. However, the set of LDCs in the ISO/DCR and of linguistic terms in OntoTag's ontologies is expected to grow in the coming years, so it is very important to keep them both linked.

On the one hand, it must not be forgotten that, *so far*, the LDCs in the ISO/DCR (1) *cannot be considered standard LDCs, since no stable process for their evaluation and standardisation has been implemented yet; and (2) cannot be viewed as a complete and/or closed set of linguistic categories*, since the eventual sets of LDCs associated to the ISO standard drafts for linguistic annotation will have to complete the addition of their corresponding LDCs by the time they become real standards.

On the other hand, ISO standardisation projects and proposals do not include and do not plan to include yet, for example, co-reference and/or anaphoric annotations and/or most forms of pragmatic annotation. These other types of linguistic annotation are expected to be formalised shortly into OntoTag's ontologies. Thus, this continued effort mentioned above should also contemplate the other way around: the new linguistic terms added to OntoTag's ontologies should also be proposed for their inclusion as standard LDCs in the ISO/DCR.

Then, *since the ISO/DCR does not focus much on the interrelations between the LDCs, if continuously linked to ISO/DCR, OntoTag's ontologies could be used as a complementary resource that makes such interrelations explicit.*

Should the first task be successfully accomplished, the **second task** to tackle would be evident: **extending also OntoTag's annotation scheme and architecture** with the new levels and categories incorporated into OntoTag's ontologies.

As for the need of supplementary research hinted by the development of OntoTagger, firstly, the impact of the linguistic tools interoperating in OntoTagger's configuration on its results should be further evaluated. That is, it should be tested (1) whether these results could be yielded by any set of similar linguistic tools interoperating in a similar way, or (2) how many OntoTagger's combination rules are tool-dependent, for example. *Such an evaluation would give an idea of the types of tools that (a) produce the best combined results when put together, or (b) interoperate more easily and why, which would also mean a practical criterion to evaluate interoperability (of which linguistic annotation is currently fairly short).*

At this point, we must remind that LACELL's POS tagger was not really included in OntoTagger's configuration. Instead, LACELL provided OEG with the annotations for a corpus sample. Therefore, the inclusion of these results in OntoTagger's configuration prevents this platform from being applied to the annotation of other corpus samples. Accordingly, LACELL's POS tagger annotations should be replaced as soon as possible by the annotations of another linguistic tool for Spanish. FreeLing²²⁷ seems to be the best candidate for this at this moment.

Secondly, new methods for the combination of linguistic annotations should be explored, in order to find out which are the most useful for the interoperation of each set of linguistic tools. For the sake of time and human resources, the only combination method implemented in OntoTagger was based on production systems. Therefore, other knowledge representation and reasoning formalisms should be implemented as well (such as Bayesian networks or statistical rules), for combining the results of the different linguistic tools interoperating in the architecture. *This could help establish if the error ratio of the combined annotations can even be further reduced.*

And thirdly, these annotation combination methods should be applied to other forms of linguistic annotation, e.g., syntactic and semantic annotation. Due to the type of annotations performed by the tools interoperating in OntoTagger, syntactic and semantic annotations could only be added to the combination of the morphosyntactic annotations, since no syntactic or semantic

²²⁷ From the UPC natural language research group, see <http://nlp.lsi.upc.edu/freeling/>.

annotations overlapped. That is, syntactic and semantic annotations were inter-level merged, but not intra-level merged. Thus, we propose here to explore these other processes of combination, either to disambiguate or to reduce the error rate of their corresponding annotations.

Finally, as far as other applicability and extensibility aspects of OntoTag are concerned, amongst others, the dependence on (A) the language (Spanish), or (B) the domain of the input texts (cinema and entertainment) and/or the domain ontology (the CNEO) featured in OntoTagger's configuration should be further researched too.

- A. *As regards the language, it would be very interesting to test if the approach of OntoTag suits the need for linguistic annotation tools of **minority languages**, for example. If it does, OntoTag could be postulated as a new way to build accurate linguistic annotation tools in these cases.*
- B. *As for the domain of the input texts and/or the domain ontology, it would be very interesting as well to find out if the same results for named entity annotation could be obtained in a different domain by simply changing the domain ontology applied and the set of language-dependent rules that implement the linguistic patterns to recognise, aggregate and subclassify them.*

Hence, also some suitable experiments to evaluate these issues should be carried out in the future.

9. ACKNOWLEDGEMENTS

The research described in this work has been supported by MCyT (Spanish Ministry of Science and Technology), MEC (Spanish Ministry for Education and Science) under the **research projects**:

1. **ContentWeb**: “Plataforma Tecnológica para la Web Semántica: Ontologías, Análisis de Lenguaje Natural y Comercio Electrónico” – TIC2001-2745 (≡ ‘Semantic Web Technologic Platform: Ontologies, Natural Language Analysis and E-Business’);
2. **Semantic Services**: “Infraestructura Tecnológica de Servicios Semánticos para la Web Semántica” – TIN2004-02660 (≡ ‘Semantic Service Technological Infrastructure for the Semantic Web’);
3. **GeoBuddies**: “Anotación Semántica Colaborativa con Dispositivos Móviles en el Camino de Santiago” – TSI2007-65677C02 (≡ ‘Collaborative Semantic Annotation with Mobile Devices in St. James' Way’);
4. **SO-CALL-ME**: “Social Ontology-driven Cognitively Augmented Language Learning Mobile Environment” – FFI 2011-29829.

It was also supported by UPM (Universidad Politécnica de Madrid) under the **research grant**:

5. **Plan-H-SemWeb**: “Plataforma de Anotación Híbrida para la Web Semántica” – Reference 14286 (A Hybrid Annotation Platform for the Semantic Web).

We must also thank **Prof. Aquilino Sánchez**, who kindly provided the OEG with the POS tagging of an ODECORPUS sample. Without these annotations, OntoTagger could not have been developed, and OntoTag could not have been evaluated.

Now, my poor English lexicon does not suffice to thank the rest of the people who has supported and encouraged me all throughout the development of this work (especialmente when the cold language of science must be abandoned to make way for the language of the heart, from where I express my thankfulness – please, excuse me, but I don’t know other way to do it). So, let me express the rest of the acknowledgements in Spanish, my mother tongue, which most of the people involved understands.

Lo primero de todo, evidentemente, tiene que ser dar las gracias a las **Guadalupe Aguado de Cea** y **Asunción Gómez Pérez**, directoras de esta tesis, por vuestro apoyo incondicional y vuestra paciencia. Una investigación que tarda sus buenos once años en completarse acabaría con la paciencia de cualquiera. Habéis sido acicate, mensaje de ánimo, a veces también de consuelo, así como un claro ejemplo de honestidad, trabajo, buen hacer y constancia. Habéis estado ahí en los altos y los bajos, aportando vuestros conocimientos y experiencia y exigiendo siempre lo mejor de mí. Sois mis madres

y mis padres en el mundo de la ciencia y la investigación y, aunque pueda no parecerlo, llevo vuestro sello en mi dura cerviz. Espero poder demostrarlo algún día. En particular, mil gracias a ti, **Asun**, por tu claridad de ideas y por poner las bases y la infraestructura para que esta tesis fuera posible; y millones de gracias a ti, **Lupe**, por tu cercanía y confianza en mí, por tu mano izquierda, y por buscarme las vueltas en medio de mis cabezonerías y avatares personales. Siempre tendré en cuenta tus consejos de redacción. Intentaré ser breve, conciso... e ir al grano... y abandonar las decimononerías... y los incisivos que distraen... y/o las parentizaciones innecesarias... y tantos otros defectos (siempre que me sea posible, claro ;-) :-D). También he aprendido de ti a apretar los dientes y a mirar al frente. Conociendo tu gran facilidad para pasar por alto lo malo y quedarte con lo bueno, ya habrás olvidado a qué me refiero. No sé muy bien si es apropiado decir esto, y menos dejarlo por escrito, pero ahí va: gracias por ser como eres, porque a muy pocas personas he conocido en mi vida que sean tan «buena gente» como tú (las comillas latinas no son casuales... ¡Vaya, ya se me ha escapado otro paréntesis innecesario! ☺).

Muchas gracias a los importantísimos *investigadores* que, de una manera u otra, me han animado a completar este largo proceso (y a quienes tanto respeto y admiro). Ya veis que vuestros tirones de orejas y vuestras ‘collejas’ (físicas, figuradas y/o virtuales) no han caído en saco roto. En este momento me acuerdo muy especialmente de **Ricardo Mairal**, **M^a Teresa Cabré**, **Mercè Lorente** y **Marie-Claude L’Homme**...

Precisamente a **Marie-Claude L’Homme** y a **Alain Polguère** tengo que agradecerles también una muy productiva estancia en la Universidad de Montreal, que supuso un importante punto de inflexión para esta tesis y en mis líneas de investigación.

Y es aquí donde tengo que expresar a **Socorro Bernardos**, **M^a Auxiliadora Barrios**, **Margarita Zozaya** e **Inmaculada Álvarez de Mon (Inma)** mi más sincero y profundo agradecimiento. Vosotras hicisteis posible que la estancia en Montreal no fuera solo productiva, sino también entretenida y divertida. Dijimos que haríamos algo así cuando yo rematara este monstruo: ahora ya no tenéis excusa para no hacerlo ;-) :-D.

A ti, **Inma**, tengo que agradecerte además toda su ayuda en muchos aspectos de esta tesis y con todo lo que de ella ha derivado y derivará, especialmente con sus publicaciones y comunicaciones asociadas. Tus correcciones y aportaciones fueron una ayuda inestimable, y siempre han mejorado ostensiblemente el resultado final. No sabes cómo lamento que nuestra colaboración esté en punto muerto. Ojalá pueda retomarse de nuevo en el futuro (y no muy lejano).

Y, por supuesto, hablando de colaboraciones, correcciones y aportaciones a la tesis y a las publicaciones, un gran MUCHÍSIMAS GRACIAS a **Charo Plaza**. Estoy convencido de que, sin ti y sin tu ayuda desinteresada, la mayor parte de mis artículos y comunicaciones no habrían sido

aceptados. Además, tus comentarios fueron muy útiles para pulir el inglés de esta memoria. El problema que tengo en este momento es que no encuentro las palabras para dar las gracias como conviene a alguien capaz de tragarse el ‘puro infumable’ de una tesis de 800 páginas (como era ésta en su primera versión ;-)). Así que no sólo te doy las gracias, sino que además te mando desde aquí un fortísimo abrazo (para que veas su alcance, es el único que figurará en los agradecimientos ;-)). Te guardo un profundo respeto y, al igual que me ocurre con Inma, me da mucha pena que las circunstancias no nos permitan vernos más a menudo.

Estos agradecimientos no estarían completos sin los que les corresponden a

- todos los miembros pasados y presentes del **Grupo de Ingeniería Ontológica (OEG)** de la UPM; en particular, debo mencionar a
 - **Javier Arrizabalaga** y **Anabel Serradilla**, pues vosotros hicisteis posible la evaluación de OntoTag con OntoTagger;
 - **Óscar Corcho**, sin quien las implementaciones de los esquemas de anotación de OntoTagger probablemente no habrían visto jamás la luz;
 - **María del Carmen Suárez-Figueroa**, por compartir conmigo tus conocimientos sobre patrones ontológicos en Marraquech (LREC 2008));
 - **Elena Montiel**, porque hablando contigo en los congresos en los que hemos coincidido he podido vislumbrar algunos de mis fallos y defectos (de los que tú careces) y eso me ha hecho reflexionar y crecer. La ciencia necesita gente como tú ;-).
 - **Ana Ibarrola** y **José Ángel Ramos**, por estar siempre al quite.
- los miembros del **grupo de investigación consolidado ATLAS (Artificial Intelligence Techniques for Linguistic ApplicationS; nº ref.: 87H31)**, de la UNED, pues formando ya parte del mismo oficialmente se terminó de revisar y escribir esta memoria. Debo mencionar, más en concreto, a **Elena Bárcena** (directora del grupo), a **Timothy Read** y a **Covadonga Rodrigo** (siempre has sido más una amiga que una colega, ya desde los tiempos de la Nebrija ;-)). La posibilidad de colaborar con vosotros ha sido un gran aliciente para acelerar la lectura de la tesis.
- a **José Luis Sierra**, director del **ILSA (Ingeniería de Lenguajes Software y Aplicaciones)** de la UCM, por poner su confianza en mí a la hora de crear su grupo de investigación.

No puedo cerrar este apartado de los agradecimientos sin dar las gracias a un sinnúmero de colegas profesores e investigadores que me han acompañado en toda o parte de esta andadura. En primer lugar, a todos los compañeros de la **Universidad Antonio de Nebrija**, a los del **Österreichisches Forschungsinstitut für Artificial Intelligence (ÖFAI, Austria)**, y a los del **Instituto de Investigación Tecnológica (IIT, Universidad Pontificia Comillas)**; en segundo lugar, a todos los **compañeros del DSIC** y a todos los **excompañeros del DSIP** de la **Universidad Complutense de**

Madrid; en tercer lugar, y también en orden alfabético, a todos aquellos que han sido fuente de aliento en algún momento: **Adriana Bolívar, Alicia Barrasa, Ángel Morillo, Blanca Gil, Christian Chiarcos, Chus Nieto, Martha Shiro,**... y tantos otros.

En el ámbito más personal, evidentemente, son muchos los *amigos* que me han acompañado y sufrido por y con mi tesis (no en vano, son 11 años de amigos ☺). Mención especial merecen (en orden alfabético, y perdón si se me olvida alguno) **Alberto García, Ángel Pradana, Daniel Pradana, Franco del Panta, Gema Sánchez, Hugo Perroni, José Luis Barco, Lucrecia Turnes, Manuel Gámez (Gamersindo), Milko Vaccaro, Olga Crespo, Paloma Pérez, Paloma Segovia, Pedro Galán, Pilar Pina, Ricardo Laserna, Roberto Martínez, Santiago Velasco, Soledad Pradana, Theodore de Souza** (sin tu generosidad, el ejemplo de los determinativos no habría sido posible), **Tim Pacífica,**... Como veis, ‘la p*ta’ (‘la p*ta tesis’, quiero decir) va bien de salud, pero no queda más remedio que rematarla :-D. Gracias a todos los excompañeros de la **Licenciatura en Informática** y “sus respectivas” (**Carlos Garoz, Carlos Marco (Tato), Cris (Belén), Elvira Becerril, Eva Morales, Fernando Ayuso, Inés González, Javier Delgado, Javier Martín, y Rocío Vázquez**) y a los compañeros del **C.P. Menéndez Pelayo** y del **I.N.B. Cervantes**. Gracias a **Mark Zuckerberg** y a **Facebook** por devolverme el contacto con estos últimos y por ayudarnos a juntarnos de nuevo. Gracias a mis más recientes amigos de **AEV**. Y gracias también a todos aquellos que han quedado por el camino, cansados de esperar a que la tesis y mis otras cienmil obligaciones dejaran un ratito libre para quedar y hacer algo juntos. Todos habéis sido compañeros de camino: gracias por los momentos compartidos.

En cuanto a los *familiares*, en fin, qué queréis que os diga: a vosotros os ha tocado lidiar con los enfados, los cabreos, los malos modos y las frustraciones del día a día de una tesis, pues muchos son los sinsabores y pocas las alegrías que conlleva. **Papá, mamá,** os doy las gracias por vuestra abnegación y por vuestras renunciadas, por vuestros consejos y vuestros silencios, por vuestros cuidados y desvelos; sin la educación que me habéis dado y vuestro buen hacer, no habría llegado este momento. *A vosotros y sólo a vosotros va dedicada mi tesis, esperando que sea satisfacción y recompensa suficiente por todo lo que os haya hecho padecer hasta aquí.* Os recuerdo que tenéis que seguir yendo al ambulatorio cuando os encontréis mal, porque no seré de ese tipo de doctores que curan enfermedades ;-):-D. Gracias por todo lo que habéis hecho por mí durante toda mi vida. **Paquita,** tú has sido la más desatendida y perjudicada en todo el proceso, porque ‘donde hay confianza da asco’, así que mil perdones y mil gracias por tu comprensión. **Mari Celes,** gracias por tu apoyo en la distancia, y a pesar de todo lo que tú llevas encima. **Marco Antonio y familia,** habéis sido mi descanso, mi reposo y mi cayado cuando ya, al final de este proceso, las fuerzas realmente flaqueaban. Gracias por regalarme mi primera ‘tesis’ una Navidad hace unos años ☺ (colgada está en casa, en el corcho del despacho). Gracias también por las ‘vacaciones’ (con el ordenador a cuestas), las bodas, celebraciones y demás ratos que hemos pasado juntos, sobre todo de risas y juegos, porque han

hecho este trabajo más llevadero. Y, por último, pedir humildemente perdón a **todos mis tíos y primos**, por haber estado ausente de tantas reuniones familiares, agradeciéndooos que nunca me lo hayáis tenido en cuenta. Espero que en algún momento podamos despertar las relaciones que se han quedado a estas alturas medio dormidas.

Back to English, I want to finish this dissertation as it started, that is, quoting (and thanking) **Prof. Charles J. Fillmore** who, pointing to a poster showing OntoTag's ontologies, said: 'Somebody has been working a lot'. St. Paul said in his First Epistle to the Thessalonians: 'Question everything; keep what is good' [1 Thes 5, 21]. I have spent too much time questioning everything in this thesis. And this is what I will keep from it: the hard working. Besides, this thesis has kept me humiliated for a number of years, and must be thanked also for that, since there is nothing more negative and harmful for students than a proud or arrogant teacher.

What could have been but was not, will not be; and 'what could have happened if...', if it has to happen, it will happen eventually. So, no more questioning, no more worrying: **this is the end of this work. Thank heaven, tomorrow will be another day and will bring another work to be done.**

10. REFERENCES

- AeroDAML (2002) Available online at <http://ubot.lockheedmartin.com/ubot/hotdaml/aerodaml.html> (visited on 06/12/2002).
- Agirre, E., Edmonds, P. (eds.) (2006) *Word Sense Disambiguation: Algorithms and Applications*. Dordrecht: Springer.
- Aguado de Cea, G., Álvarez de Mon, I., Gómez-Pérez, A., Pareja-Lora, A., Plaza-Arteche, R. (2002) 'A Semantic Web Page Linguistic Annotation Model'. *Semantic Web Meets Language Resources – Technical Report WS-02-16*. American Association for Artificial Intelligence. Menlo Park (California): AAAI Press, pp. 20–29.
- Aguado de Cea, G., Álvarez de Mon Rego, I., Pareja-Lora, A. (2009) 'Una visión interdisciplinar de la anotación semántica'. *Terminología y Sociedad del Conocimiento*, pp. 219-254. Bern, Berlin, Bruxelles, Frankfurt am Main, New York, Oxford, Wien: Peter Lang.
- Aguado de Cea, G., Pareja-Lora, A. (2000) 'A competition model for the generation of complementation patterns in machine translation'. In U. S. Bahri (ed.). *International Journal of Translation*, Vol. 12, Nº 1-2, Jan-Dec 2000. New Delhi: Bahri Publications, pp. 25-37.
- Arrizabalaga-Hernández, Francisco Javier (2004) *OntoTagger: Herramienta de anotación lingüístico-ontológica* [OntoTagger: A linguistic and ontological tool]. M.Sc. Thesis, Universidad Politécnica de Madrid.
- Barrios, M. A. (2008) 'Propuesta de descomposición semántica de fórmulas rutinarias del español en el marco de la Teoría Sentido-Texto'. In Carmen Mellado Blanco (ed.). *Colocaciones y fraseología en los diccionarios*, pp. 211-231. Frankfurt am Main: Peter Lang Internationaler, Verlag der Wissenschaften.
- Bateman, J. A. (1990) 'Upper Modeling: A General Organization of Knowledge for Natural Language Processing'. *Proceedings of the 5th International Language Generation Workshop*, Pittsburg, USA.
- Bechhofer, S., Carr, L., Goble, C., Kampa, S. and Miles-Board, T. (2002) 'The Semantics of Semantic Annotation'. *ODBASE: First International Conference on Ontologies, Databases, and Applications of Semantics for Large Scale Information Systems*. Irvine, California.

- Benjamins, V.R., Fensel, D., Decker, S., Gómez-Pérez, A. (1999) '(KA)²: Building Ontologies for the Internet: a Mid Term Report'. *IJHCS, International Journal of Human Computer Studies*, 51: 687–712.
- Bernardos-Galindo, S. (1997) *GUME: Extensión de la Ontología GUM para el Español*. [GUME: The extensión of the GUM ontology to Spanish]. M.Sc. Thesis, Universidad Politécnica de Madrid.
- Berners-Lee, T., Fischetti, M. (1999) *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web, by its Inventor*. San Francisco: Harper.
- Berners-Lee, T., Hendler, J. and Lassila, O. (2001) 'The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities'. *Scientific American*. May, 2001. <http://www.sciam.com/>
- Borst, W. N. (1997) *Construction of Engineering Ontologies*. PhD thesis, University of Twente, Enschede.
- Bray, T., Paoli, J., Sperberg, C. (1998) *Extensible Markup Language (XML) 1.0*. W3C Recommendation. <http://www.w3.org/TR/REC-xml>
- Brickley, D., Guha, R.V. (2000) *Resource Description Framework (RDF) Schema Specification*. W3C Candidate Recommendation. <http://www.w3.org/TR/PR-rdf-schema>.
- Buyko, Ekaterine, Christian Chiarcos and Antonio Pareja-Lora (2008) 'Ontology-Based Interface Specifications for an NLP pipeline architecture'. *Proceedings of the 6th Language Resources and Evaluation Conference (LREC 2008)*. Marrakech, Morocco.
- Caballero-Roldán, R., Hortalá-González, T., Martí-Oliet, N., Nieva-Soto, S., Pareja-Lora, A. y Rodríguez-Artalejo, M. (2007) *Matemática Discreta para Informáticos: Ejercicios resueltos*. Madrid: Pearson Educación, S.A.
- Calzolari, N., Corazzari, O., Zampolli, A. (2001) 'Lexical-Semantic Tagging of an Italian Corpus'. In Gelbukh, A. (ed.) *Proceedings of Computational Linguistics and Intelligent Text Processing: Second International Conference CICLing 2001*. Mexico City.
- Cantais-Sánchez, J. L. (2004) *Recubrimiento en Java de recursos lingüísticos*. [A linguistic resource Wrapping in Java]. M.Sc. Thesis, Universidad Politécnica de Madrid.
- Chinchor, N. (1997) MUC-7 *Named entity Task Definition Version 3.5*. [Available online at http://www.icl.pku.edu.cn/bswen/nlp/www.muc.saic.com/ne_task.html (visited on 17/01/2012)]

- Civit Torruella, M. (2003) *Criterios de etiquetación y desambiguación morfosintáctica de corpus en español*. Sociedad Española para el Procesamiento del Lenguaje Natural. Alicante: Universidad de Alicante.
- COHSE (2002) Available online at <http://cohse.semanticweb.org/> (visited on 06/12/2002).
- Collier, M., Manley, B. (1998) *How to read Egyptian hieroglyphs: a step-by-step guide to teach yourself*. Great Britain: British Museum Press.
- Connexor (2012) Available online at <http://www.connexor.com> (visited on 17/01/2012).
- Crystal, D. (1992) *A Dictionary of Linguistics and Phonetics* (3rd edition). Oxford: Blackwell.
- Dik, S.C. (1989) *The Theory of Functional Grammar*. Dordrecht: Foris Publications.
- Doddington, G. Mitchell, A., Przybocki, M. Ramshaw, L. Strassel, S. & Weischedel, R. (2004) 'The Automatic Content Extraction (ACE) Program. Tasks, Data, and Evaluation'. *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'2004)*. Lisbon, Portugal.
- Donoghue, C. (1999;2002) *The Mystery of the Hieroglyphs: The Story of the Rosetta Stone and the Race to Decipher Egyptian Hieroglyphs*. USA: Oxford University Press.
- Doran, P., Tamma, V., L.Iannone, L. (2007) 'Ontology module extraction for ontology reuse: an ontology engineering perspective'. *Proceedings of the ACM Sixteenth Conference on Information and Knowledge Management (CIKM 2007)*. Lisbon, Portugal.
- Downing, A., Locke, P. (2002) *A University Course in English Grammar*. London: Routledge.
- EAGLES (1996a) *EAGLES: Recommendations for the Morphosyntactic Annotation of Corpora*. EUROPEAN PROJECT DELIVERABLE: EAGLES Document EAG--TCWG—MAC/R, EAGLES Consortium, 1996. [Available online at <http://www.ilc.cnr.it/EAGLES96/annotate/annotate.html> (visited on 17/01/2012)]
- EAGLES (1996b) *EAGLES: Recommendations for the Syntactic Annotation of Corpora*. EUROPEAN PROJECT DELIVERABLE: EAGLES Document EAG–TCWG–SASG/1.8, EAGLES Consortium, 1996. [Available at <http://www.ilc.cnr.it/EAGLES96/segsasg1/segsasg1.html> (visited on 17/01/2012)]
- EAGLES (1999) *EAGLES: Preliminary Recommendations on Semantic Encoding*, Final Report. EUROPEAN PROJECT DELIVERABLE: EAGLES LE3-4244, The EAGLES Lexicon Interest

Group, 1999. [Available online at <http://www.ilc.cnr.it/EAGLES/EAGLESLE.PDF> (visited on 17/01/2012)]

Fillmore, C. J. (1968) 'The case for case'. In Emmon W. Bach & Robert T. Harms, (eds.) *Universals in Linguistic Theory*. New York: Holt, Rinehart & Winston, pp. 1–88.

Fillmore, C. J. , Atkins, S. B. (1992) 'Toward a frame-based lexicon: The semantics of RISK and its neighbors'. In Lehrer, A., Kittay, E. (eds.) *Frame, fields, and contrasts: New essays in semantic and lexical organization*. Hillsdale: Lawrence Erlbaum Associates, pp. 75–102.

Frege, F. L. G. (1892) 'Über Sinn und Bedeutung'. *Zeitschrift für Philosophie und philosophische Kritik* 100, pp. 22–50. Trad. fr. Claude Imbert, «Sens et dénotation», in *Écrits logiques et philosophiques*. Paris:Seuil, 1971, pp. 102–126.

GDO (2003) *Gran Diccionario Oxford: español-inglés, inglés-español* (2003) Oxford: Oxford University Press.

Gildea, D. & Jurafsky, D. (2002) 'Automatic Labelling of Semantic Roles'. *Computational Linguistics* 28:3, pp. 245–288.

Gildea, D. & Jurafsky, D. (2003) 'Identifying semantic relations in text'. *Exploring Artificial Intelligence in the New Millenium*, San Francisco, Morgan Kauffmann Publishers, pp. 69–102.

GOLD (2010) Available online at [<http://linguistics-ontology.org/>] (visited on 09/05/ 2010).

Gómez-Pérez, A., Corcho, O (2002) 'Ontology languages for the Semantic Web'. *IEEE Intelligent Systems and their Applications*, vol. 17(1), January/February 2002, pp. 54–60.

Gómez-Pérez, A., Fernández-López, M., Corcho, O. (2004) *Ontological Engineering: With Examples from the Areas of Knowledge Management, E-Commerce and the Semantic Web*. London: Springer-Verlag.

Greenbaum, S., Quirk, R. (1990) *A Student's Grammar of the English Language*. London: Longman.

Gruber, T. R. (1993) 'A Translation Approach to Portable Ontology Specifications'. *Knowledge Acquisition (Special Issue: Current issues in knowledge modeling)*. Vol. 5(2): 199-220.

Guiraud, P. (1969) *La Semántica*. Buenos Aires:Fondo de Cultura Económica (2nd edition).

Guiraud, P. (1966; 1955) 'La sémantique'. *Que sais-je?* series, n° 655. Paris:PUF.

- Halliday, M. A. K. (1994;1996) *An Introduction to Functional Grammar* (2nd edition). London: Arnold.
- Horrocks, I., Fensel, D., Harmelen, F., Decker, S., Erdmann, M, Klein, M. (2000) 'OIL in a Nutshell'. *12th International Conference in Knowledge Engineering and Knowledge Management (Lecture Notes in Artificial Intelligence)*, pp. 1–16. Berlin: Springer-Verlag.
<http://www.cs.vu.nl/~ontoknow/oil/download/oilnutshell.pdf>
- Horrocks, I., van Harmelen, F. (2001) *Reference description of the DAML+OIL ontology markup language*. Draft report. <http://www.daml.org/2000/12/reference.html>
- Ide, N. & Véronis, J. (1998) 'Introduction to the Special Issue on Word Sense Disambiguation: the State of the Art'. *Computational Linguistics* 24 (1), pp. 1–40.
- Ide, N., Romary, L. (2003). Outline of the International Standard Linguistic Annotation Framework. *Proceedings of ACL'03 Workshop on Linguistic Annotation: Getting the Model Right*, pp. 1–5. Sapporo, Japan.
- Ide, N., Romary, L., de la Clergerie, E. (2003). International Standard for a Linguistic Annotation Framework. *Proceedings of HLT-NAACL'03 Workshop on The Software Engineering and Architecture of Language Technology*, Edmonton, Canada.
- ISO/DCR (2008) *INTERNATIONAL STANDARD DRAFT ISO/DIS 12620.2: Specification of data categories and management of a Data Category Registry for language resources*. International Organization for Standardization (ISO): ISO/TC 37/SC3 – Terminology and other language and content resources.
- ISO/LAF (2009) *INTERNATIONAL STANDARD DRAFT ISO/DIS 24612: Linguistic annotation framework (LAF)*. International Organization for Standardization (ISO): ISO / TC 37 / SC 4 – Language resource management.
- ISO/MAF (2008) *INTERNATIONAL STANDARD DRAFT ISO/DIS 24611: Morpho-syntactic annotation framework (MAF)*. International Organization for Standardization (ISO): ISO / TC 37 / SC 4 – Language resource management.
- ISO/SemAF-NE (2009) *NEW PROJECT PROPOSAL: ISO/PWI 24167-3: Semantic annotation framework (SemAF) – Part 3: Named entities*. International Organization for Standardization (ISO): ISO / TC 37 / SC 4 – Language resource management.

- ISO/SemAF-Time (2009) *INTERNATIONAL STANDARD DRAFT: ISO/DIS 24617-1: Semantic annotation framework (SemAF) – Part 1: Time & events*. International Organization for Standardization (ISO): ISO / TC 37 / SC 4 – Language resource management.
- ISO/SynAF (2010) *INTERNATIONAL STANDARD (FINAL DRAFT): ISO/FDIS 24615:2010: Syntactic annotation framework (SynAF)*. International Organization for Standardization (ISO): ISO / TC 37 / SC 4 – Language resource management.
- Johnson, C. & Fillmore, C. J. (2000) ‘The FrameNet tagset for frame-semantic and syntactic coding of predicate-argument structure’. *Proceedings of ANLP-NAACL’2000*. Seattle, USA.
- Jurafsky, D. & Martin, J. H. (2000) *Speech and Language processing*. New Jersey: Prentice Hall.
- Karp, R., Chaudhri, V., Thomere, J. (1999) *XOL: An XML-Based Ontology Exchange Language*. Technical Report. <http://www.ai.sri.com/~pkarp/xol/xol.html>
- Kent, R. (1998) *Conceptual Knowledge Markup Language (version 0.2)*. Available online at <http://sern.ucalgary.ca/KSI/KAW/KAW99/papers/Kent1/CKML.pdf>.
- Kietz, J-U., Maedche, A., Volz, R. (2000) ‘A Method for Semi-Automatic Ontology Acquisition from a Corporate Intranet’. *Proceedings of the EKAW 2000 Workshop on Ontologies and Text*. Juan-les-Pines, France. October, 2000.
- Kilgarriff, A. & Rosenzweig, J. (2000) ‘English SENSEVAL: Report and Results’. *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC 2000)*. Athens, Greece. <ftp://ftp.itri.bton.ac.uk/reports/ITRI-00-25.ps.gz>
- Kilgarriff, A. (1998) SENSEVAL: ‘An Exercise in Evaluating Word Sense Disambiguation Programs’. *Proceedings of the 1st International Conference on Language Resources and Evaluation (LREC’98)*, pp. 581–588. Granada, Spain. <ftp://ftp.itri.bton.ac.uk/reports/ITRI-98-09.ps.gz>
- Kingsbury, P., Palmer, M. and Marcus, M. (2002) ‘Adding Semantic Annotation to the Penn TreeBank’. *Proceedings of the Human Language Technology Conference (HLTC 2002)*. San Diego, USA.
- Kiryakov, A., Popov, B., Terziev, I., Manov, D., Ognyanoff, D. (2005) ‘Semantic Annotation, Indexing, and Retrieval’. *Elsevier’s Journal of Web Semantics*, Vol. 2(1).

- Kokkinakis, D. (2004) 'Reducing the effect of name explosion'. *Proceedings of the LREC'2004 Workshop 'Beyond Named Entity Recognition – Semantic Labelling for NLP tasks'*, pp. 1–6. Lisbon, Portugal.
- Kokkinakis, D., Kokkinakis, S. J. (1999) *Sense-Tagging at the Cycle-Level using GLDB*. (Research Reports from the Department of Swedish, GU-ISS-96-5). Göteborg:Göteborg University, Sweden.
- Lassila, O., Swick, R. (1999) *Resource Description Framework (RDF) Model and Syntax Specification*. W3C Recommendation. <http://www.w3.org/TR/PR-rdf-syntax>
- Lázaro, F., Tusón, V. (1990) *Lengua española*. Madrid: Editorial Anaya.
- Leech, G. (1997) 'Introducing corpus annotation'. In Garside R., Leech, G., McEnery, A. M. (eds.) *Corpus Annotation: Linguistic Information from Computer Text Corpora*. London: Longman.
- Lowe J., Baker C., Fillmore C. (1997) 'A frame-semantic approach to semantic annotation'. *Proceedings of the ACL Special Interest Group on the Lexicon (SIGLEX) Workshop 'Tagging Text with Lexical Semantics: Why, What, and How?'*. Association for Computational Linguistics. Washington D.C., USA, pp. 8–24.
- Luke S., Heflin J. (2000) *SHOE 1.01. Proposed Specification*. SHOE Project. <http://www.cs.umd.edu/projects/plus/SHOE/spec1.01.htm>
- Lyons, J. (1977) *Semantics*. Cambridge: Cambridge University Press.
- Mann, W. & Thomson, S. (1988) *Rhetorical Structure Theory: Toward a functional theory of text organization*. *Text* Vol.18, 3: 243–281.
- Maynard, D., Bontcheva, K., & Cunningham, H. (2003) 'Towards a semantic extraction of named entities'. *Proceedings of Recent Advances in Natural Language (RANLP 2003)*, Bulgaria.
- McEnery, A. (2003) 'Corpus Linguistics'. In Mitkov, R. (ed.) *The Oxford Handbook of Computational Linguistics*. Oxford: Oxford University Press.
- McEnery, A. M., Wilson, A. (2001) *Corpus Linguistics: An Introduction*. Edinburgh: Edinburgh University Press.
- McEnery, A., Xiao, R., and Tono, Y. (2006) *Corpus-Based Language Studies*. New York: Routledge.

- McGuinness, D.L., van Harmelen, F. 2003. *OWL: Web Ontology Language Overview* (W3C Candidate Recommendation). <http://www.w3.org/TR/owl-features/>
- Mel'čuk, I. (1996) 'Lexical Functions: A Tool for the Description of Lexical Relations in the Lexicon'. In L. Wanner (ed.): *Lexical Functions in Lexicography and Natural Language Processing*, pp. 37-102. Amsterdam/Philadelphia: Benjamins.
- Merriam-Webster (2012) Merriam-Webster Online Dictionary [available online at <http://www.merriam-webster.com> (visited on 18/01/2012)]
- Motta, E., Buckingham Shum, S. Domingue, J. (1999) 'Case Studies in Ontology-Driven Document Enrichment'. *Proceedings of the 12th Knowledge Acquisition Workshop*, Banff, Alberta, Canada.
- Narayanan, S., Petruck, R. L., Baker, C. F., Fillmore, C. J. (2003) 'Putting FrameNet Data into the ISO Linguistic Annotation Framework'. *Proceedings of the ACL' 2003 Workshop on Linguistic Annotation: Getting the Model Right*. Sapporo, Japan.
- Navarro, B., Civit, M., Martí, M.A., Marcos, R., Fernández, B. (2003) 'Syntactic, Semantic and Pragmatic Annotation in Cast3LB'. In Schumacher, F. (ed.) *Proceedings of the Workshop on Shallow Processing of Large Corpora (SProLaC 2003)*. Lancaster, UK.
- Nirenburg, S., Raskin, V. (2004) *Ontological Semantics*. Massachusetts: MIT Press.
- OALD (2005) *Oxford Advanced Learner's Dictionary* (7th edition). Oxford : Oxford University Press.
- OntoMat (2002) Available online at <http://annotation.semanticweb.org/ontomat.html> (visited on 06/12/2002).
- OntoWeb (2002) *OntoWeb: The Thematic Network for the Semantic Web*. [Available online at : <http://www.ontoweb.org/> (visited on 06/12/2002)]
- Pino, M. & Santalla, M. P. (1996) 'Codificación de la Anotación Morfosintáctica en SGML'. *Revista de la Sociedad Española para el Procesamiento del Lenguaje Natural (SEPLN)*, 19(1996):101–117. [Available on line at <http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/3751/2209> (visited on 17/01/2012)].
- RosettaNet (2002) *RosettaNet: Lingua Franca for eBusiness*. <http://www.rosettanet.org/> (visited on 06/12/2002).

- Ruimy, N., Corazzari, O., Gola, E., Spanu, A., Calzolari, N., Zampolli, A. (1998) 'The European LE-PAROLE Project: the Italian Syntactic Lexicon'. Proceedings of the 1st International Conference on Language Resources and Evaluation, pp. 241-248. Granada, Spain.
- Ruppenhoffer, J. Baker, C.F. & Fillmore, Ch. (2002) 'Collocational Information in the FrameNet Database'. In Braasch, A. & Poulen, C. (eds) *Proceedings of the 10th Euralex International Congress*. Copenhagen. Denmark.
- Sagüés-Subijana, M. (1980). *Gramática elemental vasca: gramática comparada*. San Sebastián: Txertoa.
- Sanfilippo, A., Tratz, S., Gregory, M., Chappell, A., Whitney, P., Posse, C., Paulson, P., Baddeley, B., Hohimer, R., White A. (2006) 'Automating Ontological Annotation with WordNet'. *Proceedings of the Third International WordNet Conference*. Jeju Island, Korea.
- Saussure, Ferdinand de (1916/1983) *Course in General Linguistics*, trans. by Roy Harris (1983). London: Duckworth.
- Schmidt, K. M. (1988) 'Der Beitrag der begriffsorientierten Lexicographie zur systematischen Erfassung von Sprachwandel und das Begriffswörterbuch zur mhd. Epik'. In Bachofer, W. (ed.) *Mittelhochdeutsches Wörterbuch in der Diskussion*, pp. 35–49. Tübingen: Max Niemeyer.
- Serradilla-Fernández, A. I. (2004) Integración de recursos semánticos y aprendizaje de named entities en OntoTagger. [Semantic resource integration and named entity learning in OntoTagger] M.Sc. Thesis, Universidad Politécnica de Madrid.
- SFN (2012) *Spanish FrameNet: An On-Line Lexical Resource and its Application to Spanish NLP*. [Available online at <http://gemini.uab.es:9080/SFNsite> (visited on 17/01/2012)]
- Shea, D. (2005) Who Cares about Semantics Anyway? [Available online at http://www.mezzoblue.com/archives/2005/05/30/who_cares_ab/ (visited on 05/08/2005)]
- SHOE (2002) <http://www.cs.umd.edu/projects/plus/SHOE/KnowledgeAnnotator.html> (visited on 06/12/2002).
- SIMPLE (2000) *The SIMPLE Project*. <http://www.ub.es/gilcub/SIMPLE/simple.html> (visited on 06/12/2002).
- Staab, S., Angele, J., Decker, S., Erdmann, M., Hotho, A., Mädche, A., Schnurr, H.-P., Studer, R. (2000) 'Semantic Community Web Portals'. *Proceedings of the 9th International World Wide Web Conference on Computer Networks*. Amsterdam, The Netherlands.

- Stevenson, M. & R. Gaizauskas (2000) 'Using Corpus-derived Name Lists for Named Entity Recognition'. *Proceedings of the 6th Conference on Applied NLP and 1st Conference of the North American Chapter of the ACL*. Seattle, USA.
- Stevenson, M. & Wilks, Y. (2003). 'Word-sense disambiguation'. In Mitkov, R. (ed.) *The Oxford Handbook of Computational Linguistics*. Oxford: Oxford University Press.
- Stubbs, M. (1996) *Text and Corpus Analysis*. Oxford: Blackwell.
- Studer, R., Benjamins, R. & Fensel, D. (1998) 'Knowledge Engineering: Principles and Methods'. *Data and Knowledge Engineering (DKE)*, Vol. 25, 1-2, pp. 161–197.
- Suárez, A., Palomar, M. (2002) 'Word Domain vs. Word Sense Disambiguation: a Maximum Entropy Approach'. *Lecture Notes in Computer Science*, vol. 2448, pp. 131-138. Springer-Verlag.
- Subirats, C. (2004). 'Automatic processing of textual information in Spanish'. *Jornada de Seguimiento de Proyectos en Tecnologías Informáticas*. Málaga (Spain).
- Tapanainen, P., Järvinen, T. (1997). "A Non-Projective Dependency Parser". *Proceedings of the 5th Conference on Applied Natural Language Processing (ANLP)*. Washington, D.C., USA. March/April, 1997.
- Trier, J. (1931) *Der deutsche Wortschatz im Sinnbezirk des Verstandes: Die Geschichte eines sprachlichen Feldes*. Band I: Heidelberg.
- UNSPSC (2002) *Universal Standard Products and Services Classification (UNSPSC)*. [Available online at <http://www.unspsc.org/> (visited on 06/12/2002).
- Uschold, M. (2003) 'Where are the Semantics in the Semantic Web?'. *AI Magazine*, 24 (3): 25–36. Menlo Park (California): American Association for Artificial Intelligence (AAAI).
- Vargas-Vera, M., Domingue, J., Motta, E., Shum, S. B., Lanzoni, M. (2001) 'Knowledge Extraction by Using an Ontology-based Annotation Tool'. *Proceedings of the Knowledge Markup & Semantic Annotation Workshop*, held in conjunction with the First International Conference on Knowledge Capture (K-CAP 2001), pp. 5–12. Victoria, Canada. October, 2001.
- Véronis, J. (2003) 'Sense tagging: does it make sense?' [Published version of a paper presented at the Corpus Linguistics'2001 Conference, Lancaster, U.K.]. In Wilson, A., Rayson, P. and McEnery, A. (eds.) *Corpus Linguistics by the Lune: a festschrift for Geoffrey Leech*. Frankfurt: Peter Lang.

- Vossen, P., Bloksma, L., Rodriquez, H., Climent, S., Calzolari, N., Roventini, A., Bertagna, F., Alonge, A., Peters, W. (1998) 'The EuroWordNet Base Concepts and Top Ontology'. *EuroWordNet (LE-4003) Deliverable D017D034D036*, University of Amsterdam.
- Wierzbicka, A. (1988) *The semantics of grammar*. Amsterdam: John Benjamins.
- Wilson, A., Thomas, J. (1997) 'Semantic Annotation'. In R. Garside, G. Leech & A. M. McEnery (eds.) *Corpus Annotation: Linguistic Information from Computer Text Corpora*. London: Longman.
- WordReference.com (2012) <http://www.wordreference.com> [visited on 18/01/2012]
- Wright, S. E. (2004) 'A Global Data Category Registry for Interoperable Language Resources'. *Proceedings of the 4th Language Resources and Evaluation Conference (LREC 2004)*. Lisbon, Portugal [Available online at <http://www.lrec-conf.org/proceedings/lrec2004/pdf/87.pdf> (visited on 17/01/2012)]
- Zauzich, K. T. (1992; 2004) *Hieroglyphs without mystery: an introduction to ancient Egyptian writing*. Japan: University of Texas Press.

11. APPENDIX A: ONTOTAGGER'S ANNOTATION XML SCHEMA

The XML code associated to OntoTagger's XML Schema, used for the generation and validation of the final annotation documents outputted by OntoTagger, is shown in Table 149.

Table 149: OntoTagger's XML Schema

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema" elementFormDefault="qualified">
  <xsd:element name="aspect">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:pattern value="lvo:\{([I,P])\{([I,P])*\}\|lvo:[I,P]0"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="auxiliary">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:pattern value="lvo:\{([B,H])\{([B,H])*\}\|lvo:[B,H]0"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="case">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:pattern
value="lvo:\{([A,C,D,E,G,I,L,N,O,P,S,V])\{([A,C,D,E,G,I,L,N,O,P,S,V])*\}\|lvo:[A,C,D,E,G,I,L,N,O,P,S,V]0"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="definiteness">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:pattern value="lvo:\{([D,I,U])\{([D,I,U])*\}\|lvo:[D,I,U]0"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element name="degree">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:pattern value="lvo:\{([C,P,S])\{([C,P,S])*\}\|lvo:[C,P,S]0"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:complexType name="dependencyType">
    <xsd:simpleContent>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="ad"/>
        <xsd:enumeration value="ada"/>
        <xsd:enumeration value="ads"/>
        <xsd:enumeration value="agt"/>
        <xsd:enumeration value="cc"/>
        <xsd:enumeration value="cla"/>
        <xsd:enumeration value="cnt"/>
        <xsd:enumeration value="com"/>
        <xsd:enumeration value="comp"/>
        <xsd:enumeration value="dat"/>
        <xsd:enumeration value="det"/>
        <xsd:enumeration value="goa"/>
        <xsd:enumeration value="ha"/>
        <xsd:enumeration value="loc"/>
        <xsd:enumeration value="main"/>
      </xsd:restriction>
    </xsd:simpleContent>
  </xsd:complexType>

```

```

        <xsd:enumeration value="man"/>
        <xsd:enumeration value="mod"/>
        <xsd:enumeration value="neg"/>
        <xsd:enumeration value="obj"/>
        <xsd:enumeration value="phr"/>
        <xsd:enumeration value="pm"/>
        <xsd:enumeration value="qn"/>
        <xsd:enumeration value="subj"/>
        <xsd:enumeration value="tmp"/>
        <xsd:enumeration value="v-ch"/>
        <xsd:attribute name="head" use="required">
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN"/>
            </xsd:simpleType>
        </xsd:attribute>
    </xsd:restriction>
</xsd:simpleContent>
</xsd:complexType>
<xsd:element name="finiteness">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="lvo:\{([F,N]\|[F,N])*\}|lvo:[F,N]|0"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="gender">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="lvo:\{([F,M,N,C]\|[F,M,N,C])*\}|lvo:[F,M,N,C]|0"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="inflection_type">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="lvo:\{([M,S,W]\|[M,S,W])*\}|lvo:[M,S,W]|0"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="lemma" type="xsd:string"/>
<xsd:complexType name="morphoType">
    <xsd:all>
        <xsd:element ref="aspect" minOccurs="0"/>
        <xsd:element ref="auxiliary" minOccurs="0"/>
        <xsd:element ref="case" minOccurs="0"/>
        <xsd:element ref="definiteness" minOccurs="0"/>
        <xsd:element ref="degree" minOccurs="0"/>
        <xsd:element ref="finiteness" minOccurs="0"/>
        <xsd:element ref="gender" minOccurs="0"/>
        <xsd:element ref="inflection_type" minOccurs="0"/>
        <xsd:element ref="number" minOccurs="0"/>
        <xsd:element ref="person" minOccurs="0"/>
        <xsd:element ref="politeness" minOccurs="0"/>
        <xsd:element ref="possessive_number" minOccurs="0"/>
        <xsd:element ref="reflexivity" minOccurs="0"/>
        <xsd:element ref="separability" minOccurs="0"/>
        <xsd:element ref="strength" minOccurs="0"/>
        <xsd:element ref="tense" minOccurs="0"/>
        <xsd:element ref="verb_form_mood" minOccurs="0"/>
        <xsd:element ref="voice" minOccurs="0"/>
    </xsd:all>
    <xsd:attribute name="category" type="xsd:string" use="required"/>
</xsd:complexType>
<xsd:element name="morpho-syntactic_function">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="lvo:AJ"/>
            <xsd:enumeration value="lvo:AP"/>
            <xsd:enumeration value="lvo:AT"/>
            <xsd:enumeration value="lvo:AV"/>
            <xsd:enumeration value="lvo:C"/>
            <xsd:enumeration value="lvo:N"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>

```



```

        <xsd:enumeration value="lvo:NU"/>
        <xsd:enumeration value="lvo:PD"/>
        <xsd:enumeration value="lvo:R"/>
        <xsd:enumeration value="lvo:V"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element name="number">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="lvo:\{([P,S]\{([P,S])*\})\}|lvo:[P,S]0"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:complexType name="paragraphType">
    <xsd:sequence>
        <xsd:element ref="sentence" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="id" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN"/>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:complexType>
<xsd:element name="person">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="lvo:\{([1,2,3]\{([1,2,3])*\})\}|lvo:[1,2,3]0"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="phrase_function">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="lvo:\{([H,P,R]\{([H,P,R])*\})\}|lvo:[H,P,R,0]"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="politeness">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="lvo:\{([P,F]\{([P,F])*\})\}|lvo:[P,F]0"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="possessive_number">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="lvo:\{([P,S]\{([P,S])*\})\}|lvo:[P,S]0"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element name="Annotated_Document" type="Annotated_documentType"/>
<xsd:element name="reflexivity">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="lvo:\{([N,R]\{([N,R])*\})\}|lvo:[N,R]0"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:complexType name="semanticsType">
    <xsd:all>
        <xsd:element ref="use" minOccurs="0"/>
        <xsd:element ref="participant_type" minOccurs="0"/>
        <xsd:element ref="word_meaning" minOccurs="0"/>
        <xsd:element ref="is_related_to" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:all>
    <xsd:attribute name="category" type="xsd:string" use="required"/>
</xsd:complexType>
<xsd:complexType name="sentenceType">
    <xsd:sequence>
        <xsd:element ref="token" maxOccurs="unbounded"/>
    </xsd:sequence>

```

```

</xsd:sequence>
<xsd:attribute name="id" use="required">
  <xsd:simpleType>
    <xsd:restriction base="xsd:NMTOKEN"/>
  </xsd:simpleType>
</xsd:attribute>
</xsd:complexType>
<xsd:element name="separability">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string"/>
  </xsd:simpleType>
</xsd:element>
<xsd:element name="strength">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="lvo:\{([W,S])\{([W,S])*\}\|lvo:[W,S]0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element name="surface_tag">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="lvo:\{([A,C,D,N,P,Q,R,V])\{([A,C,D,N,P,Q,R,V])*\}\|lvo:[A,C,D,N,P,Q,R,V]0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element name="syntactic_function">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="lvo:\{([D,I,P,S,U,0])\{([D,I,P,S,U,0])*\}\|lvo:[D,I,P,S,U]0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:complexType name="syntaxType">
  <xsd:all>
    <xsd:element ref="depends_syntactically_on" minOccurs="0"/>
    <xsd:element ref="morpho-syntactic_function" minOccurs="0"/>
    <xsd:element ref="phrase_function" minOccurs="0"/>
    <xsd:element ref="surface_tag" minOccurs="0"/>
    <xsd:element ref="syntactic_function" minOccurs="0"/>
  </xsd:all>
  <xsd:attribute name="category" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="lvo:TS"/>
        <xsd:enumeration value="lvo:TM"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>
<xsd:element name="tense">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="lvo:\{([I,P,R,F])\{([I,P,R,F])*\}\|lvo:[I,P,R,F]0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element name="text" type="xsd:string"/>
<xsd:complexType name="tokenType">
  <xsd:sequence>
    <xsd:element ref="syntax" minOccurs="0"/>
    <xsd:element ref="discourse" minOccurs="0"/>
    <xsd:element ref="word" maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="id" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:NMTOKEN"/>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="DataLexica_id" type="xsd:NMTOKEN" use="required"/>
  <xsd:attribute name="FDG_id" type="xsd:NMTOKEN" use="required"/>
  <xsd:attribute name="UMurcia_id" type="xsd:NMTOKEN" use="required"/>

```

```

</xsd:complexType>
<xsd:element name="verb_form_mood">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="lvo:\{([C,F,G,I,N,P,S,U])\{([C,F,G,I,N,P,S,U])*\}\}lvo:[C,F,G,I,N,P,S,U]0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element name="voice">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="lvo:\{([A,P])\{([A,P])*\}\}lvo:[A,P]0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:complexType name="wordType">
  <xsd:sequence>
    <xsd:element ref="text"/>
    <xsd:element ref="lemma"/>
    <xsd:element ref="syntax"/>
    <xsd:element ref="morpho"/>
    <xsd:element ref="semantics" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="id" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:NMTOKEN"/>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="guessed" type="xsd:boolean" use="optional"/>
</xsd:complexType>
<xsd:element name="use">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="lvo:\{([A,P])\{([A,P])*\}\}lvo:[A,P]0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element name="participant_type">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="lvo:\{([A,G,B,N,P,S,T,R,O,V,C,U,I,D,E])\{([A,G,B,N,P,S,T,R,O,V,C,U,I,D,E])*\}\}0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element name="paragraph" type="paragraphType"/>
<xsd:complexType name="Annotated_documentType">
  <xsd:sequence>
    <xsd:element ref="paragraph" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name="sentence" type="sentenceType"/>
<xsd:element name="token" type="tokenType"/>
<xsd:element name="syntax" type="syntaxType"/>
<xsd:element name="word" type="wordType"/>
<xsd:element name="morpho" type="morphoType"/>
<xsd:element name="semantics" type="semanticsType"/>
<xsd:element name="depends_syntactically_on" type="dependencyType"/>
<xsd:complexType name="wordMeaningType">
  <xsd:sequence>
    <xsd:element ref="word_sense" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name="word_sense">
  <xsd:complexType>
    <xsd:attribute name="resource" type="xsd:string" use="required"/>
    <xsd:attribute name="term_id" type="xsd:string" use="required"/>
  </xsd:complexType>
</xsd:element>
<xsd:element name="word_meaning" type="wordMeaningType"/>
<xsd:complexType name="is_Related_ToType">
  <xsd:sequence>
    <xsd:element ref="unit" minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>

```

```

        </xsd:sequence>
        <xsd:attribute name="rel_type" type="xsd:string" use="required"/>
    </xsd:complexType>
    <xsd:element name="is_related_to" type="is_Related_ToType"/>
    <xsd:element name="synonymy" type="xsd:string"/>
    <xsd:element name="hyperonymy" type="xsd:string"/>
    <xsd:element name="meronymy" type="xsd:string"/>
    <xsd:element name="holonymy" type="xsd:string"/>
    <xsd:complexType name="unitType">
        <xsd:sequence>
            <xsd:element ref="synonymy" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="holonymy" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="hyperonymy" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="meronymy" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute name="term_id" type="xsd:string" use="required"/>
        <xsd:attribute name="lang" type="xsd:string" use="required"/>
        <xsd:attribute name="type" type="xsd:string" use="optional"/>
    </xsd:complexType>
    <xsd:element name="unit" type="unitType"/>
    <xsd:complexType name="discourseType">
        <xsd:attribute name="category" type="xsd:string" use="required"/>
        <xsd:attribute name="domain" type="xsd:string" use="required"/>
        <xsd:attribute name="subcategory" type="xsd:string" use="required"/>
        <xsd:attribute name="MUC-7_tag" type="xsd:string" use="required"/>
    </xsd:complexType>
    <xsd:element name="discourse" type="discourseType"/>
</xsd:schema>

```

In this XML Schema, some patterns have been created to describe the concrete possible values that each attribute can take (and to prevent any others from being assigned to it). These patterns, notated in by means of regular expressions, allow for the concurrent assignation of several possible values to an attribute, implementing thus the ambiguity denotation and underspecification mechanism presented at the beginning of Section 4.3, with the format also explained in that section: [*value*₁ | *value*₂ | ... | *value*_n].

The value ‘0’ (character zero) has been included into each and one of the particular sets of possible values which can be taken by each attribute, in order to represent the cases where that attribute is not applicable to a specific unit (the attribute *tense* is not applicable to *nouns*, for example and, therefore, it is assigned a ‘0’ value). By way of example, the pattern describing the values of the attribute *gender*, $\backslash[(F,M,N,C)\backslash[(F,M,N,C)]^*\backslash[F,M,N,C,0]$, can be described as follows:

- Values between brackets, separated by commas, are a type of choice list, i.e., one and only one of the different values consigned can be chosen at a time. This notation stands for a low-level disjunction notation.
- The character ‘|’ is used to signal a higher level disjunction notation, i.e., the disjunction of two regular expressions which represent mutually exclusive value (sub)patterns which can be assigned to a given attribute, but not concurrently.
- When the occurrence of a bracket (‘[’, ‘]’) or a ‘|’ character (the disjunction symbol in the notation) is required in the consignment of a value (for expressing an underspecification,

for example), it has to be shown that they are part of the pattern itself by the escape character ‘\’ put in front of any of these three characters.

- Parenthesis are used to group or associate regular (sub)expressions in a given pattern. This makes it easier to express that a (sub)expression can be repeated any number (from 0 to ∞) of times, which is notated by means of the character ‘*’ put immediately after the (sub)expression which can be repeated.