COMPLEX NATURAL LANGUAGE PROCESSING SYSTEM ARCHITECTURE

Stefan DIACONESCU[#], Ionut DUMITRASCU[*]

[*] SOFTWIN, Bucharest Romania

Corresponding author: Stefan DIACONESCU

The paper presents the main components of a complex Natural Language Processing System from the implementation point of view. Based on certain linguistic theoretical background, the system contains a Grammar Abstract Language (GRAALAN) that allows by its sections the description of some linguistic chapters: alphabet, syllabification, morphology, inflection rules, inflection forms, lexicon, syntax, bilingual lexical correspondences, bilingual morphological correspondences, and bilingual syntactic correspondences. A set of tools (like the GRAALAN compiler, inflection forms creation tool, lexicon creation tool, bilingual linguistic correspondences tool) helps the linguist to create a Linguistic Knowledge Base (LKB) that contains information about languages and about the correspondence between languages. The LKB is in XML format and therefore is compatible with a set of XML DTDs, one for each corresponding section from GRAALAN. Different linguistic applications can be developed using the knowledge from the LKB: morphological analyzers, grammar checkers, inflection applications, indexing/searching applications, lemmatizers, spellers, hyphenators, lexicons, lexical dictionaries, human assisted / computer assisted / automatic machine translation applications.

Key words: natural language processing; linguistic knowledge base; machine translation;

1. INTRODUCTION

Many NLP systems were conceived and implemented during the last almost half century. Usually these systems concerned only some more wide or more narrow sections of informatic linguistics. Seldom some broad fields waere tackled during large projects like EUROTRA [1], EAGLES [6], ROSETTA [15]. Unfortunately, these large projects have not materialised in successful implementations, but they have the great worth of pushing forward at least the theoretical study of the domain. One of the major drawbacks (among others) was the lack of unity among different linguistic chapters approach. Paradoxically, this lack of unity has grown for the worse due to the (successful) standardisation effort of the different linguistic chapters representation, because the extremely useful approach of each individual section was not sufficiently correlated with the approach of other linguistic sections.

The paper presents some basic architecture elements that belong to complex aproaches and that try to realise some unity between different linguistic chapters, namely: alphabet, syllabification, inflection rules, inflection forms, lexicon, morphology, syntax, bilingual correspondences. This unity is realised using a language (named GRAALAN - Grammar Abstract Language) that can be used to describe the aforesaid linguistic chapters (see Fig. 1). The most important elements of this language are based on a few theoretical concepts: GDG - Generative Dependency Grammar, DT - Dependency Tree, AVT - Attribute Value Tree (section 2). Starting from these basic concepts, GRAALAN allows the creation of the above mentioned linguistic information (section 3). The creation of the linguistic knowledge is supported by some informatic tools (section 4). Using these tools, a linguistic knowledge base can be created for individual languages or for pairs of languages (section 5). The linguistic knowledge bases allow the development of different linguistic applications (section 6). Finally, the section 7 presents some results concerning the implementation of this system.

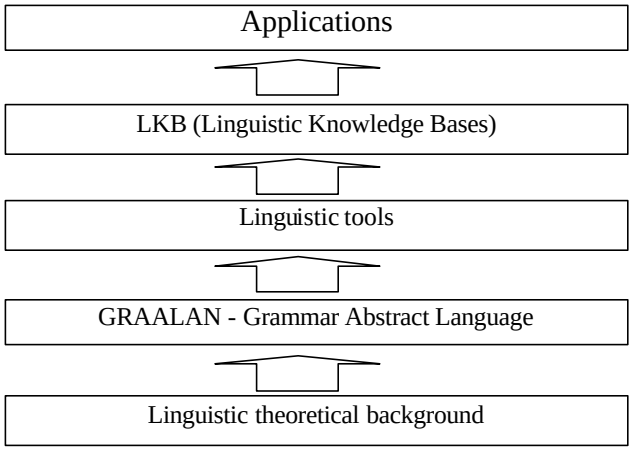

Fig. 1 NLP System components

2. BASIC THEORETICAL CONCEPTS

2.1. ATTRIBUTE VALUE TREES

The AVT - Attribute value trees are used to describe lexical and/or syntactic categories and their values under the form of a tree. Such a tree can (simplified and informally) be described as follows (in a BNF like formalism)[3]:

1. <AVT>	->	{<label> : <attribute list>} | {<label>} | {<attribute list>} | <attribute list>
2. <attribute list>	->	<attribute><attribute list> | <attribute>
3. <attribute>	->	[<attribute content>]
4. <attribute content>	->	<label>: <feature content> | <feature content> | <label>
5. <feature content>	->	<attribute name> = <attribute value list>
6. <attribute value list>	->	<attribute value element>, <attribute value list> | <attribute value element>
7. <attribute value element>	->	<attribute value name> <AVT> | <attribute value name>

The <attribute name>, <attribute value name> and <label> are character strings different from ":,". The labels are used to define and use subtrees. A more formal definition is given in [3]. The AVT are used in GRAALAN to describe: inflection rules, inflection forms, morphology, syntax.

2.2. DEPENDENCY TREES

The dependency trees DT (see Fig. 2) are used to represent the structure of a phrase [1]. A DT is an 6-tupple *DT = {N, T, P, A, SR, CR}* where:

- *N* is the set of non-terminals *n* i.e. the set of syntactic categories that can be described having a name and a structure. A non-terminal can be decomposed in others elements from *N, T, P, A, SR, C*R.
- *T* is the set of the terminals *t* i.e. the set of words that can be found in the lexicon or can be obtained by applying some flexional rules on words from the lexicon.
- *P* is the set of pseudo-terminals *p* i.e. the set of non-terminals that contain only terminals. When we will describe a dependency tree or a grammar we will not cover all the words from the lexicon because in this case the

number of rules from the grammar will be too big. So we can say that some non-terminals that we name pseudo-terminals (for example some nouns or some verbs) will never be described in the grammar.

- *A* is the set of procedural actions *a* i.e. the set of the routines that can be used to represent a certain portion of the text that we analyze. For example a number represented like a sequence of digits or a mathematical formula or even an image with a certain significance that appears in a text can be "replaced" in grammars or dependency trees by a certain procedural action.

- *SR* is the set of subordinate relations *sr* i.e. the set of the relations between *N, T, P, A, C*R, respecting some rules. The links that enter in an *sr* come from one element that is considered to be subordinated to the elements that receive links that comes from this *s*r.

- *CR* is the set of coordinate relations *cr* i.e. the set of the relations between *N, T, P, A, SR,* respecting some rules. The links that enter in a *cr* come from some elements that are considered to be coordinated to each other but also from some other elements. The links that come from the coordinated elements (usually 2) are named fixed entries. The other entries are named supplementary entries.

The DT is used in GRAALAN to describe syntax, some parts of lexicon (concerning the MWE - Multiword Expressions [4], for example), the inflection rules and the correspondences between structures of two languages.

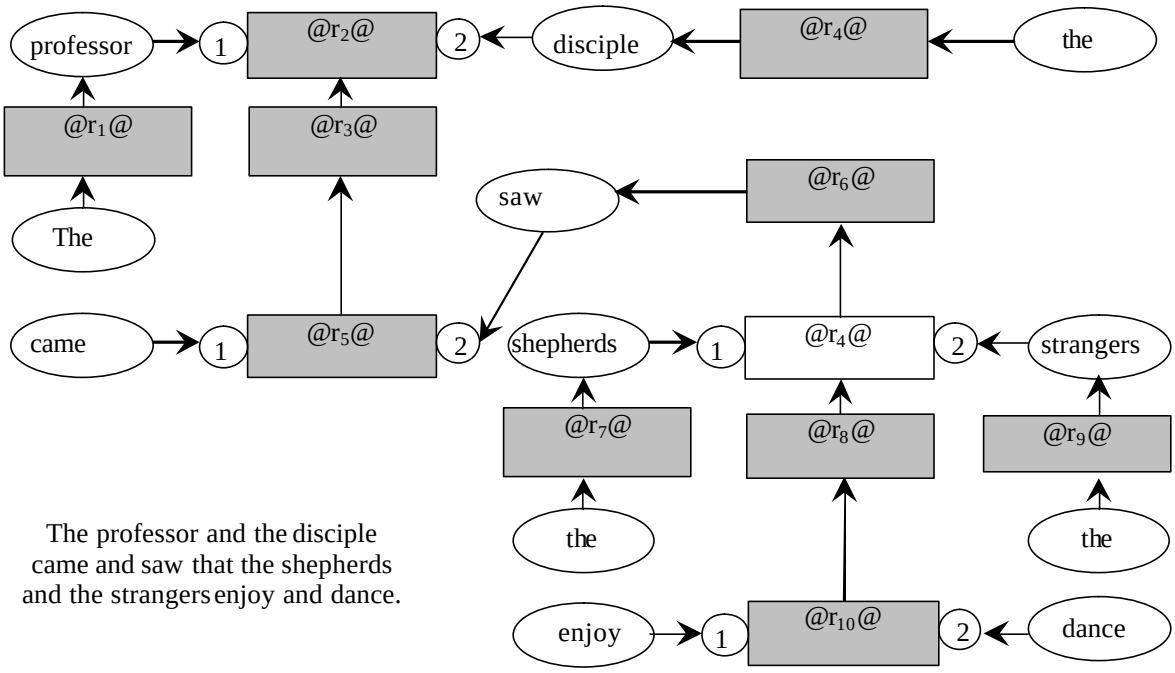

Fig. 2 Example of DT

2.3. GENERATIVE DEPENDENCY GRAMMAR

The generative dependency grammar GDG is used to generate a phrase and the structure of this phrase [2][5]. A generative dependency grammar GDG is an 8-tupple $GDG = \{N, T, P, A, SR, CR, n_0, R\}$ where:

- *N, T, P, A, SR, CR* was defined in the section 2.2.
- n_0 belongs to N and is named root symbol.
- *R* is a set of numbered rules of the form *(i)* $n_i \rightarrow (s_i, q_i)$ where; n_i belongs to N; s_i is a sequence of elements from $N \cup T \cup P \cup A$ (we will note also *ntpa* such an element), q_i is a dependency tree having nodes from s_i and oriented labels from $CR \cup SR$ and $i = 1, 2, 3, \ldots$.

GDG has the important property that, in the generation process, it builds not only the string (that must be recognised in the input for example), but also the associated dependency tree, giving therefore a structure to an unstructured text.

Using GDG and AVT we can define also a GDGF - GDG with Features as follws [1]: a GDG with feature structure is a GDG where each *ntpa* can have associated an AVT.

The GDGF is used in GRAALAN to describe the syntax.

3. GRAALAN - GRAMMAR ABSTRACT LANGUAGE

3.1. ALPHABET

Different codes associated to the signs used to graphicaly represent a language can be expressed in GRAALAN section Alphabet. The following codes can specified:

a) The phonetic alphabet used to describe the language. It can be for example a subset of IPA (International Phonetic Alphabet) [13].

b) The normal alphabet used to write the language. In fact, it can be not only an usual alphabet but also some ideograms or ideographes like chinese characters.

c) Special characters used for the language representation

d) Groups of characters that represent the association between some normal alphabet codes and the corresponding phonetic codes (for example diftongs, triftongs in Romanian language or the pronunciation of the ideograms).

e) Alphabetic classes that can be for example the class of vowels, the class of consonants etc.

What is defined in GRAALAN Alphabet section can be used in other GRAALAN sections that use some terminals (words, characters, morphems, etc.)

All the characters are considered to use UNICODE [10].

Example of an alphabet fragment for the Romanian language:

Phonetic alphabet
/* a */ ***character code*** = "a" ***type*** = ***internal label*** = open_central_unrounded
/* ? */ ***character code*** = "ə" ***type*** = ***internal label*** = mid_central_unrounded
.....
Normal alphabet
/* a */ ***character code*** = "a" ***type*** = ***internal label*** = a
/* A */ ***character code*** = "A" ***type*** = ***internal label*** = A
.....
Groups
/* a, A */ ***group code*** = (("a", "A") ["&open_central_unrounded;"]) ***label*** = a_group
/* iou */ ***group code*** = (("iI"/"oO"/"uU") [("&semivowel_i;&mid_back_rounded;&semivowel_u;")])
label = triphthong_iou
.....
Classes
class label = vowel ***elements*** = ("a", "A", "e", "E", "i", "I", "o", "O", "u", "U", "ă", ...)
class label = diphthong
elements = ("&diphthong_ai;", "&diphthong_au;", "&diphthong_ei;", ...)
.....

3.2. SYLLABIFICATION

GRAALAN considers three types of syllabification:

a) Euphonical syllabification: it referes to the words written in the normal (eventually in special) alphabet and observing the pronunciation mode.

b) Phonetic syllabification: it referes to the words written with phonetic alphabet and observing also the pronunciation mode.

c) The morphological syllabification: it is an analogue to the euphonical syllabification but it must also observe some supplementary restrictions that involve the morphematic structure of the word.

To each of these syllabification types we can add some supplementary restrictions concerning the end of written lines (*the hyphenation*).

The elements that can be considered in the syllabification process are:

a) the characters of the normal alphabet.

b) Groups (diftongues, triftongues, etc.) described in the GRAALAN Alphabet section.

c) Some special characters: apostrophes, hypen, etc.

d) Some constituent elements (morphemes) defined in the lexicon.

There are two types of sylllabification rules corresponding to the first two types of syllabifications that must be defined in GRAALAN Syllabification section: euphonic syllabification rules and phonetic syllabification rules. These two types of rules can be very close in the case of languages like Romanian or Russian, but they can be quite different in languages like French, very different in the languages like English and extremely different in the languages like Chinese, that depend on the writing systems. The morphological syllabification has no special rules defined in GRAALAN Syllabification section because it respects the implicit rules derived from the morphems that are present in the lexicon.

Example of a few euphonic syllabification rules for the Romanian language:

Euphonic

/* 1 */ ***Rule*** "&vowel;" - "&vowel;";
/* 2 */ ***Rule*** "&vowel;" - "&diphthong;";
/* 3 */ ***Rule*** "&vowel;" - "&triphthong;";
/* 4 */ ***Rule*** "&diphthong;" - "&diphthong;";
/* 5 */ ***Rule*** "&vowel;" - "&consonant;" + "&vowel;";
/* 5.1 */ ***Rule*** "&vowel;" - "&che_cons_voc;";
.....

3.3. MORPHOLOGY

The morphology of a language (more exactly the set of lexical categories and their values) are presented in GRAALAN like an AVT tree where the attribute node type corresponds to lexical categories and the value node type corresponds to the attribute value. These two types of nodes have also some supplementary information attached:

a) The attribute nodes contain:

- The lexical category name;
- The abbreviation of the lexical category name;
- The indication if the category is inflected or not (corresponding to its position in the morphological tree);
- (Eventually) the name of a program (procedural action) that can be used to associate some specific treatement in the current point of the tree.

b) The value nodes contain:

- The lexical category name;
- The abbreviation of the lexical category name;
- The indication if the corresponding form is a lemma or not;
- The indication if the corresponding form is present as a main entry in the lexicon (lemma), as a secondary input in the lexicon, or is not present in the lexicon at all.

In GRAALAN morphological section it can be also indicated what are the inflected situations that correspond to identical inflected forms.

Example of a fragment of morphology description for the Romanian language:

Section Morphological Configurator

Tree

[clasa / ***name*** = Clasa, ***abbreviation*** = Cls, ***inflection*** = no /
= substantiv / ***name*** = Substantiv, ***abbreviation*** = Subst, ***lemma*** = yes, ***lexicon*** = input /
[tip substantiv / ***name*** = TipSubstantiv, ***abbreviation*** = TipSubst, ***inflection*** = no /
= comun / ***name*** = Comun, ***abbreviation*** = Com, ***lemma*** = yes, ***lexicon*** = input /
, propriu / ***name*** = Propriu, ***abbreviation*** = Pr, ***lemma*** = yes, ***lexicon*** = input /]
[animatie / ***name*** = Animatie, ***abbreviation*** = Animat, ***inflection*** = no /
= animat / ***name*** = Animat, ***abbreviation*** = Anim, ***lemma*** = yes, ***lexicon*** = input /

, inanimat / ***name*** = Inanimat, ***abbreviation*** = Inanim, ***lemma*** = yes, ***lexicon*** = input /]
[GEN_NEFLEXIONAT: ***gen*** / ***name*** = Gen, ***abbreviation*** = Gen, ***inflection*** = no /
= masculin / ***name*** = Masculin, ***abbreviation*** = Masc, ***lemma*** = yes, ***lexicon*** = input /
, feminin / ***name*** = Feminin, ***abbreviation*** = Fem, ***lemma*** = yes, ***lexicon*** = input /
, neutru / ***name*** = Neutru, ***abbreviation*** = Neu, ***lemma*** = yes, ***lexicon*** = input /]
.....]

3.4. INFLECTION RULES

A lexicon entry that can be inflected (a lemma for example) identifies a compound inflection rule in GRAALAN inflection rules section. A compound rule is a list of basic inflection rules. A basic rule is in fact an attribute value type tree that indicates more inflection situations, one for each tree leaf. Each inflection situation (i.e. a leaf) has one or more elementary inflection rules associated with it. An elementary inflection rule contains:

a) A condition (logical expression) that points out when the elementary inflection rule can be applied.

b) A transformation operations list (insert, add, delete) that must be executed on lemma (or on other inflected form) in order to obtain the current inflected form, expressed in normal alphabet.

c) Analogue with (b) but corresponding to the phonetic alphabet.

d) In the case of an analytical or synthetical-analytical form: an AVT for each component word and the relations between these component words (practically this means the dependency tree associated to the correponding inflection form).

Based on inflection rules from the GRAALAN inflection rules section, we can obtain all the inflection forms from the GRAALAN inflection forms section.

Example of a fragment of a basic inflection rule (only alphabetic forms):

Basic Rule Subst_masc1: [clasa = substantiv] [tip substantiv = comun] [tip substaniv comun = animat, inanimat] [gen = masculin] [numar = singular [caz = nominativ [articulare =
nearticulat (EtS0: ***alphabetic*** -),
hotarat (EtS11: ***if***(&consonant;) ***alphabetic*** insert "ul"
if("i") ***alphabetic*** insert "ul"
if("u") ***alphabetic*** insert "l"
if("e") ***alphabetic*** insert "le"),
nehotarat (EtS12: alphabetic insert word left "un" [clasa = articol] [tip articol = nehotarat] [caz = nominativ] [gen = masculin] [numar = singular] @acord_gen-numar-caz@)
],
genitiv .....]]

3.5. INFLECTION FORMS

GRAALAN inflection forms section contains one entry for each inflection form. An entry contains the following information:

a) The inflected form written in normal alphabet and in phonetic alphabet.

b) The identification of the lexicon entry the inflection form corresponds to.

c) The features of the inflection form under the form of a list of attributes with their values..

d) The way to make the syllabification for the inflection form: euphonic, phonetic, morphologic and the hypentation.

e) The way to sort the inflection form in a list of inflection forms. It contains what word must be used (for the forms with more that one word - analytical or synthetical-analytical forms) and the sense of the sorting (from left to right or from right to left).

The inflection forms are not usually written directly in GRAALAN by the linguist, but they are generated using a special inflection form tool.

3.6. LEXICON

A lexicon is in GRAALAN a set of entries where each entry can have one of the following types:

a) Morphemes that can be roots, prefixes, suffixes, prefixoides, suffixoides, etc.

b) Words that can be: main entries of lemma type, supplementary inputs that are associated to a lemma and some main entries that are not lemmas .

c) Multi Word Expression (MWE) that contains also the structure under the form of a dependency tree. These dependency trees indicate also the level of variability of the nodes in different instances that MWE can have: total variable, partially variable, invariable.

d) Analytical or synthetical-analytical structures (inflected forms with more that one word) that are similar with MWE where always the central (head) word is total variable.

e) Syntactic structures that are also similar to MWE but contain as nodes non-terminals and the nodes can have associated not lexical but syntactic categories too (with their values).

Depending on its type, a lexicon entry can also have many other informations associated, like:

a) Semantical information: gloss, synonims, antonyms, paronyms, hipernyms, hiponyms, meronyms, omonyms, connotations, etc.

b) Etymological information: the original language, the original form, the transliteration of the original form in the alphabet of the current language.

c) Syllabification information: euphonic syllabification, phonetic syllabification, morphological syllabification.

d) Morphological information: some lexical categories (with their values), the identification of the asociated inflection rule (that are present in the GRAALAN inflection rules section), the segmentation.

e) The sorting mode: how the corresponding entry is put in a list of sorted entries.

The lexicon is not usually written directly in GRAALAN by the linguist, but they are generated using a special lexicon tool.

3.7. SYNTAX

A language syntax is described in GRAALAN as a list of labeled syntactic rules based on the principles of GDG.

A rule has two members. The left member contains a non-terminal accompanied by an AVT formed by lexical and syntactic categories and values. The right member contains one or more alternants. An alternant has three sub sections:

a) Syntax subsection. It contains a sequence of *ntpa-s*. Each *ntpa* contains a name (the *ntpa* name), an AVT and the linking mode with other *ntpa*-s.

b) Dependency subsection. In this section, the dependency relations are described: the governor-subordinate relations and the coordination relations.

c) Agreement subsection. The agreement between different *ntpa*-s is described as a sequence of conditions like: "if (<condition expression>) true (<actions>) false (<actions>) not applicable (<actions>) undefined (<actions>)". The condition expression is a logical expression that refers some *ntpa*-s from the syntactic subsection and some of their features (lexical/syntactic categories and values). The actions indicates how the corresponding situation must be treated : error messages, the continuation mode after the error detection.

The syntax described in GRAALAN has the reversibility property i.e. it can be used both in the syntactic analyse process (through which the dependency tree is produced from the surface [input] text) and in the generation process (through witch the surface text is produced from the dependency tree).

Example of a simplified fragment of a rule for the Romanian languge syntax description (it uses some macros for agreement):

.....

Rule RegentSubordonatNominalAtributivNominativAcuzativ:

<regent/subordonat> [rol grup complex = nominal-atributiv] [caz = nominativ, acuzativ] [animatie = neanimat, animat] [gen = masculin, feminin, neutru] [numar = singular, plural] [animat = I, II, III] ::=

.....

Alternant Alternant2:

Syntax

Eticheta1: <articol> [tip articol = nehotarat] (gen = masculin, feminin, neutru) (numar = singular, plural) [caz = nominativ, acuzativ] *Subordinate* Eticheta6

Eticheta2: <secventa grup complex subordonat> [rol grup complex = nominal-atributiv] [tip = corelativ, distributiv, logic] [pozitie fata de acordant = stanga] [animatie = neanimat, animat] (gen = masculin, feminin, neutru) (numar = singular, plural) [animat = I, II, III] (articulare = nearticulat) [caz = nominativ, acuzativ] ***Subordinate*** Eticheta7

Eticheta3: <regent> [rol grup complex = nominal-atributiv] (exprimat prin = substantiv, pronume, numeral) (pozitie fata de acordant = stanga) [animatie = neanimat, animat] [gen = masculin, feminin, neutru] [numar = singular, plural] [animat = I, II, III] (articulare = nearticulat) [caz = nominativ, acuzativ] ***Governor*** Eticheta5, Eticheta6, Eticheta7

Eticheta4: <atribut> [rol grup complex = nominal-atributiv] [exprimat prin = adjectiv propriuzis] (pozitie fata de acordant = dreapta) (gen = masculin, feminin, neutru) (numar = singular, plural) (articulare = nearticulat) [caz = nominativ, acuzativ] ***Subordinate*** Eticheta5

Dependencies

Eticheta5: @relatie atribut / regent@
Eticheta6: @relatie articol nehotarat@
Eticheta7: @relatie atribut / regent@

Agreement

$regula simpla a numarului SN ("Eticheta1", "Eticheta3")$
$regula simpla a genului SG ("Eticheta1", "Eticheta3")$
$regula simpla a numarului SN ("Eticheta2", "Eticheta3")$
$regula simpla a genului SG ("Eticheta2", "Eticheta3")$
$regula simpla a numarului SN ("Eticheta4", "Eticheta3")$
$regula simpla a genului SG ("Eticheta4", "Eticheta3")$

3.8. BILINGUAL CORRESPONDENCES

The GRAALAN section regarding the bilingual correspondences contains the following types of elements belonging to two different languages:

*a) Correspondences between MWE*s - Multiple Word Expressions. MWE are represented in the lexicons as dependency trees. The correspondence express the equivalence between the source expression and the target expression and the transformation rules that indicate how the extensions of the source expression in different instances are transfered as extensions of the target expression.

b) Correspondences between the words. It is a particular case of correpondence between MWEs where both MWEs have only one word.

c) Correspondences between syntactic structures. It is a particular case of the correspondence between MWEs where the nodes of the two expression can have associated not only lexical categories and values but also syntactic categories and values.

d) Correspondences between morphological structures. It is a particular case of the correspondence between MWEs where at lesast the source expression corresponds to analytical or syntactical-analytical inflection forms.

e) Correspondences between morphological subtrees. It is a correspondence between different sets of lexical categories and values organised as AVTs.

The informations from the GRAALAN bilingual correspondences section are used in the generation of some dictionaries and in machine translation applications.

3.9. MACROS AND MESSAGES

GRAALAN offers two other useful features: the macros and the messages.

a) *The macros* system allows a more compact writing of the GRAALAN text sequences that are identical or very close to one another. This involves the existence of a macroprocessor that handles the macros and generates the pure GRAALAN text (see Fig. 3).

b) *The messages* system allows the entire system using messages from different languages. In this way, the linguistic knowledge base can be accessed from aplication user interfaces written for different natural languages.

4. LINGUISTIC TOOLS

4.1. INFLECTION FORM TOOLS

The inflection forms of the words that are in the lexicon can be automatically generated using the inflection rules. In fact, for all the natural languages, there are a lot of exceptions from a certain set of inflection rules that are considered in a classical grammar. A special tool will help the linguist to generate, verify and correct the inflection forms. This tool will take as input the already introduced GRAALAN inflection rules, the already generated GRAALAN inflection forms and the corrections made by the linguist and will update the GRAALAN inflection rules and the GRAALAN inflection forms. The resulting GRAALAN inflection rules and GRAALAN inflection forms will be compiled and the corresponding linguistic knowledge base is generated. In this way, the linguist will manage the inflection rules and the inflection forms using an interactive user interface.

4.2. LEXICON TOOL

The information in the lexicon can be described directly using GRAALAN but it is a tedious task. In fact, the information in the lexicon appears using a special lexicon tool that has the following main functions:

a) The introduction of morphemes and words, one by one.

b) The introduction of multiword expressions, one by one, but recursing to a more complex graphical interface and using the analyse functions based on syntax description.

c) The automatic introduction of the analytical or synthetic-analytical morphologic structures generation using the information from the inflection rules.

d) The automatic introduction of the syntactical structures using the information from the syntax description.

The lexicon tool will generate the GRAALAN text that is compiled with the GRAALAN compiler and the lexicon linguistic data base is created.

4.3. GRAALAN COMPILER

GRAALAN is a language that can be used by a linguist to specify the linguistic knowledge. In fact, as we presented in the previous section, a GRAALAN text can result also from a GRAALAN MACRO text or it can be obtained by the generation with a specific tool.

A more apropriate format for the applications is XML [17] that has also the advantage to be handled with some standard tools like DOM - Document Object Model [18]. Therefore GRAALAN text will be converted in XML using a specific compiler (see Fig. 3).

4.4. BILINGUAL CORRESPONDENCY TOOL

The bilingual correspondencies can be written directly in GRAALAN but it is more handy for the linguist to use a special tool. This tool will exploit the information from the lexicons and from the morphology description and will expose this information to the linguist, in a graphical manner. We remember that the correspondences are structural mappings between the elements of the two languages. For example, this tool will display the dependency trees for the selected multi word expressions, syntactic structures or morphologic structures. The linguist will establish the correspondences between the selected elements from the linguistic knowledge bases for the two languages and will add the needed information.

The tool will then generate automatically the corresponding GRAALAN text, will compile this text with the GRAALAN compiler and will put the results in the linguistic knowledge base.

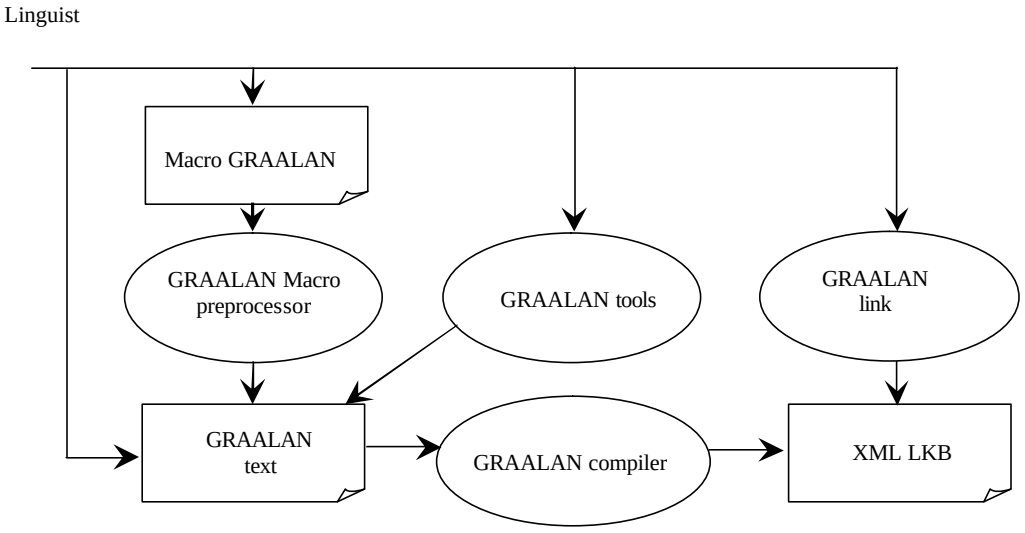

Fig. 3 Using GRAALAN

4.5. LINKER

The linguistic knowledge in GRAALAN format will consist of hundreds or perhaps thousands of files obtained on different ways: from GRAALAN text or GRAALAN macro text written by the linguist, generated by different tools like inflection form tool, lexicon tool or bilingual correspondencies tool. This files will be compiled and the corresponding XML files will be generated and put in the linguistic knowledge base. A special tool, the linker, will check the compatibility between all these XML files, for exemple: the use of the same lexical categories with the same lexical categories values, the use of the defined alphabet in all the files, etc. All the incompatibilities will be listed and the linguist will be able to correct the situations, if he deems it necessary.

5. LINGUISTIC KNOWLEDGE BASE

The linguistic knowledge base is generated by compiling the GRAALAN texts in XML. The errors are detected using the linker tool. Finally, the linguistic knowledge base wil contain knowledge about languages and pairs of languages. The XML format being a standard format used by almost all the operating systems and computer platforms, the linguistic knowledge bases are portable, i.e. they can be moved between different operating systems and computer platforms. The information in XML linguistic knowledge bases can be used by applications through DOM or other standard or specific tools.

6. APPLICATIONS

Many linguistic applications can be developed based on the information from the linguistic knowlege base: morphological analyzers, grammar checkers, inflection applications, indexing/searching applications, lemmatizers, spellers, hyphenators, lexicons, lexical dictionaries, human assisted / computer assisted / automatic machine translation applications. The machine translation applications are, of course, the most

complex among these applications and we will say a few words only about this type of application, in order to give a flavor of how differently the informations from the linguistic knowledge base could be used.

The GDG representation of the language syntax (including AVTs and DTs) allows a three step translation mechanism (see Fig. 4):

a) - The analysis: the conversion of the source text (the surface representation written in the source language L_1) to the dependency tree (the deep representation). In order to do this, the source text is first of all annotated, i.e. to each word from the source text there will be one or more interpretations attached, based on the inflected forms of the source language L_1. This will be a synthetical morphological level treatement. Using the source GDG, the annotated text is translated to the corresponding dependency tree that will continue to contain only synthetic morphological forms. The dependency tree is then transformed by replacing some of its subtrees with analytical or synthetical-analytical morphological dependency trees taken from the lexicon of the source language L_1.

b) - The dependency trees conversion: the conversion of the source dependency tree (corresponding to the source language L_1) to the target dependency tree (corresponding to the target language L_2). It is a very complex process and we will present here only few ideas about the needed operations. In this process the bilingual correspondencies will be used. The analytical or synthetical-analytical forms from the source language will be replaced with the correponding forms in the target language. The multi word expression from the source language will be replaced with the corresponding words or multi word expression in the target language. Finally, the remaining words from the source language will be replaced with the corresponding words from the target language. During this dependency trees conversion a complex desambiguation mechanism (that we do not present here) will be used. In this way we obtain a dependency tree corresponding to the target language.

c) - The generation: the conversion of the dependency tree corresponding to the target language L_2 to the surface representation corresponding to the target language L_2. First of all, the analytical or sinthetical-analytical forms are expanded in its synthetical correspondences tree using the information from L_2 lexicon. After that, using the L_2 syntax description, the target surface form in L_2 will be generated.

As we can see, in this translation precess different types of operations are used: (a) text synthetical annotation, (b) text analytical or synthetical-analytical annotation, (c) text analysis (in order to generate the dependency tree corresponding to a surface representation), (d) surface text generation (from the corresponding dependency tree representation), etc. These basic complex operations can be used in other types of applications too, for example: (a) are used in synthetical morphological analysers; (a), (b) and (c) are used in the analytical or synthetical-analytical analysers; (a) and (b) are used in grammar checkers, etc.

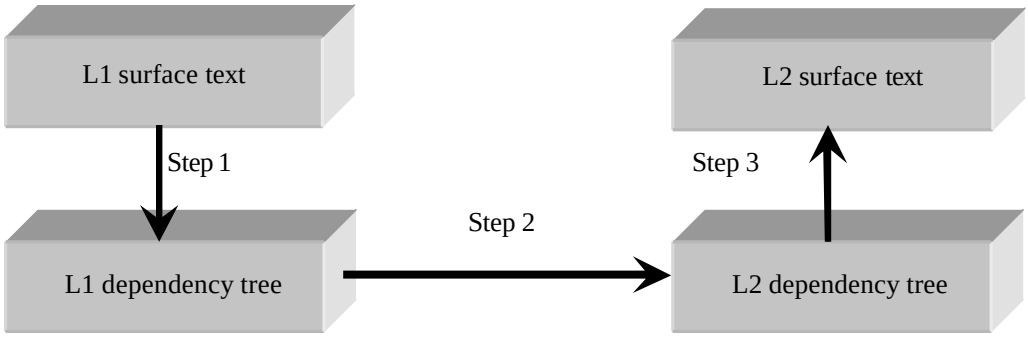

Fig. 4 Three step translation

7. CONCLUSIONS

We presented the basic architectural elements of a complex NLP system that allows the linguistic knowledge creation, representation and use in linguistic applications. The linguistic knowledge can be organised on three levels: i) the general language knowledge (alphabet, syllabification, inflection rules, morphology, syntax) for a language; ii) the specific linguistic knowledge (inflection forms, lexicon) for a language: iii) the bilingual linguistic knowledge (bilingual correspondences) for a pair of languages. First of

all, the general linguistic knowledge must be created and fully implemented using the GRAALAN compiler for each language in the system. After that, the specific linguistic knowledge (that represents information associated to morphems, words and multiword expressions) must be created step by step, enriching gradually the linguistic knowledge bases for each language in the system. When we have all the general linguistic knowledge for two languages in the linguistic knowledge base and enough specific linguistic knowledge for the two languges (at least a few tens of thousands of morphems, words and multiwords expressions) we can define also step by step the linguistic correspondences. The basic tools of the system (especially the GRAALAN compiler) have already been implemented and used to create some of the general linguistic knowledge for the Romanian languge. The specific linguistic language knowledge for the Romanian language will be created using system tools and new languages will be introduced in the sysytem. An important fact must be mentioned: the introduction of a new language in the system is accelerated when we already have some languages in the system, using a method of "work on a model", i.e. using the already introduced information as references. This is possible because the lingusitic knowledge is expressed in formal and intuitivelly accessible manner using GRAALAN.

Afterwards, the knowledge in the linguistic knowledge bases will be used not only for different linguistic applications developemnt, but also for some linguistic knowledge studies.

REFERENCES

1. ALSHAWI, H., ARNOLD, D.J., BACKOFEN, R., CARTER, D.M., LINDOP, J., NETTER, K., PULMAN, S.G., TSUJII, J., USZKOREIT, H., *EurotraET6/1: Rule Formalism and Virtual Machine Study. Final Report*, Commission of the European Communities, 1991.
2. DIACONESCU, S., *Natural Language Understanding Using Generative Dependency Grammar*, in Max Bramer, Alun Preece and Frans Coenen (Eds), ES 2002. Research and Development in Intelligent Systems XIX, Proceedings of ES2002, the Twenty second SGAI International Conference on Knowledge Based Systems and Applied Artificial Intelligence, Cambridge UK, Springer, pp.439-452, 2002
3. DIACONESCU, S., *Some Properties of the Attribute Value Trees Used for Linguistic Knowledge Representation*, in 2nd Indian International Conference on Artificial Intelligence (IICAI-05) INDIA during December 20-22, 2005
4. DIACONESCU, S., *Multiword Expression Translation Using Generative Dependency Grammar*, in Proceedings of ESTAL 2004 - ESPAÑA for NATURAL LANGUAGE PROCESSING October 20-22, Alicante, Spain, 2004
5. DIACONESCU, S., *Natural Language Agreement Description for Reversible Grammars*, in Tamás D. Gedeon, Lance Chun Che Fung (Eds.), AI 2003: Advances in Artificial Intelligence, 16th Australian Conference on AI, Perth, Australia, Proceedings, pp. 161-172, 2003
6. EAGLES, *Formalism Working Group Final Report*, Version of september 1996
7. EAGLES *Recommendations for the Morphosyntactic Annotation of Corpora*, EAG--TCWG--MAC/R, Version of Mar, 1996
8. EAGLES: *Preliminary Recommendations on Subcategorisation*, EAG---CLWG---SYNLEX/P Version of Aug, 1996
9. EAGLES: *Lexicon architecture Draft Report*, EAG--LSG/IR--T1.1, Version of Oct, 1993
10. ISO/IEC 10646:1992 Information technology -- Universal Multiple-Octet Coded Character Set (UCS) ISO/IEC 10646-1:1993
11. ISO 639:1988 *Code for the representation of names of languages*
12. ISO 639-2:1995 *Code for the representation of names of languages*
13. International Phonetic Association: *Handbook of the International Phonetic Association*, A Guide to the Use of the International Phonetic Alphabet
14. LENCIi, A., BEL, N., BUSA, F., CALZOLARI, N., GOLA, E., MONACHINI. M., OGONOWSKI, A., PETERS, I., PETERS, W., RUIMY, N., VILLEGAS, M., ZAMPOLLI, A, *SIMPLE: A General Framework for the Development of Multilingual Lexicons*
15. LANDSBERGEN, J., *Isomorphic grammars and their use in the ROSETTA translation system*, in Machine Translation Today: The State of the Art, Edinburgh University Press, Edinburgh, 1987
16. TESNIÈRE, L., *Éléments de syntaxe structurelle*, Paris, Klincksieck, 1959
17. W3C: *Extensible Markup Language (XML)* 1.0, Recommendation 10-Feb-98
18. W3C: *Document Object Model (DOM)* Level 1 Specification (Second Edition), Version 1.0, September, 2000