A Formalized Reference Grammar for UNL-based Machine Translation between English and Arabic

Sameh Alansary $^{1,2}$
(1) Bibliotheca Alexandrina, Alexandria, Egypt
(2) Faculty of Arts, Department of Phonetics and Linguistics, Alexandria University
El Shatby, Alexandria, Egypt
Sameh.alansary@bibalex.org

ABSTRACT

The Universal Networking Language (UNL) is an artificial language that can replicate human language functions in cyberspace in terms of hyper semantic networks. This paper aims to: a) design a reference grammar capable of dealing with the basic linguistic structures in order to act as a test-bed in automating translation between English and Arabic in both directions through UNL; b) evaluate the current state of the UNL system as an Interlingua in analyzing and generating English and Arabic as far as the reference structures are concerned. A reference parallel corpus of 500 structures was used. Results are promising; precision and recall of analyzing English to UNL (UNLization) are 0.979 and 0.96 respectively, while precision and recall of analyzing Arabic to UNL are 0.98 and 0.96 respectively. Precision and recall of generating English from UNL (NLization) are 0.97 and 0.96 respectively, while precision and recall of generating Arabic from UNL are 0.989 and 0.96 respectively.

KEYWORDS: Reference Grammar, Formal Grammar, Interlingua, UNL, UNL-ization Grammar, NL-ization Grammar, Machine Translation, Universal Networking Language, UNL system.
Introduction

While languages differ greatly in their “surface structures”, they all share a common “deep structure”; hence came the idea of creating a universal representation capable of conveying this deep structure while enjoying the regularity and predictability natural languages lack. Although interlingua is a promising idea, the number of interlinguas created is still very limited. Examples of well-known interlinguas are DLT (Witkam 2006), UNITRAN (Dorr 1987, 1990) and (Dorr et al. (2004)), KANT (Nyberg and Mitamura (1992), Nyberg et al. (1997)) and UNL (Uchida 1996, Uchida and Zhu (1993, 2005), Alansary et al. (2010)). The first three of these interlinguas lack standardization, however, the fourth, UNL, has succeeded in standardizing its tools, tagset and methodology as well as rely on meaning as an intermediate representation since all languages share the same meaning representation of the same sentence. A complete comparison and evaluation of the previously mentioned interlinguas can be found in Alansary (2011).

UNL is a kind of mark-up language which represents not the formatting but the core information of a text, which yielded a format for "perfectly aligned" multilingual documents. Currently, 16 languages are participant in the UNL programme. The UNDL Foundation, the founder of UNL, has created a wrapper application for development of various UNL tools and applications through the UNL development platform and a number of UNL language resources that supports the UNL system (Martins 2012, Martins and Avatesyan 2009). All engines, resources and tools are available through the UNLweb website: www.unlweb.net. These tools are designed for implementing different tasks based on UNL technology. The “Unl.dev” contains many tools designed for professional users (linguists, computational linguists) as well as non-professionals. These tools are used in analysing and generating natural languages. IAN, the Interactive ANalyzer, is a semi-automatic analysis engine that operates according to a linguistic and realistic design to analyze natural language input in order to extract meaning and represent it in terms of hyper-semantic networks. IAN works for all languages, it simply employs the analysis grammar rules to analyze input and finally generate its corresponding UNL expressions. It operates semi-automatically; word-sense disambiguation is still carried out by a language specialist, nevertheless, the system can filter the candidates using an optional set of disambiguation rules; “IAN” can be accessed online through (http://dev.undlfoundation.org/analysis/index.jsp). EUGENE (the dEep-to-sURface natural language GENERator) is a fully automatic engine; it does not involve any human intervention. Similar to IAN, EUGENE is language-independent, it simply uses the target language grammar rules in order to decode the incoming UNL document and generate it in natural language format; “EUGENE” can be accessed online through (http://dev.undlfoundation.org/generation/index.jsp). Both IAN and Eugene use two types of Natural language dictionaries; enemurative and generative. The enumerative dictionary of IAN contains all inflected word forms of a language together with their corresponding Universal Words (concepts) and a set of linguistic features covering different linguistic levels. The generative dictionary, on the other hand, is the same as the ‘enmurate’ one but it contains all lexemes of language as bases together with a morphological paradigm number that controls the generative morphological behaviour (e.g. agreement and inflected forms) of words in natural language (Martins and Avetisyan 2009). A complete description of Arabic enumerative and generative dictionaries can be found in Alansary (2012). It might be a fact that all languages have classical reference grammars in grammar books. Such a reference grammar maybe defined as a description of the grammar of a language, with explanations of the principles governing the construction of words, phrases, clauses, and sentences. It is designed to teach someone about the language and to give readers a reference tool for looking up specific details of the language. In
Natural Language Processing, computers should also learn a language in order to give a comprehensive and objective test-bed that enables us to evaluate, compare and follow up the performance of different grammars in Natural Language Processing in general and Interlinguas in particular. A formalized reference grammar is needed in order to synchronize different languages. Up to our knowledge, such an idea is quite new idea as it may be true that there is no such universal or standardized environment that link different systems. The Universal Networking Language is initiating this idea as it utilizes a standardized environment; all participant languages in the UNL system are building their grammars synchronized to each other through a specific parallel multilingual corpus; a corpus compiled to cover common structures in English and translated to all languages in action in the UNL system. In the current paper we are limiting ourselves to English and Arabic only. In this paper we are limiting ourselves to English and Arabic only. It is organized as follows: Section 1 discusses the design and compilation of the reference corpus. Section 2 discusses the design and implementation of the analysis grammar for both English and Arabic in terms of UNL. Section 3 discusses the design and implementation of the generation grammar for both English and Arabic. Section 4 evaluates the analysis and generation results in English and Arabic. And finally section 5 is a conclusion and future work.

1 Reference corpus

Corpora are considered essential language resources necessary when building grammars. A reference corpus has been compiled as an experimental English corpus in order to prepare the initial version of analysis and generation grammars. An Arabic parallel Corpus has been compiled by translating the English reference corpus into Arabic, this corpus consists of 500 sentences collected from English grammar books. It is supposed to cover the basic and common linguistic phenomena between all languages that may be encountered in the process of building grammars within the UNL framework such as: temporary entries (e.g. URLs, nonsense words, symbols etc.), words that are not found in the dictionary (a grammar in NLP may face a set of words that might not be found in the dictionary), numerals, determiners, prepositions, conjunctions, noun phrase structures, expressions of time, verb forms, pronouns and sentence structures. The English reference corpus is manually annotated to make a standard version of UNL reference corpus. Both versions; English reference and UNL corpora, are available on the UNL web (http://www.unlweb.net/wiki/Corpus500). The Arabic UNL language centre has translated the English reference corpus into Arabic. Table 1 shows the statistical division of the corpus.

| Type                  | Number of sentences |
|-----------------------|---------------------|
| Temporary entries     | 15                  |
| Numbers               | 150                 |
| Determiners           | 60                  |
| Prepositions          | 40                  |
| Conjunctions          | 40                  |
| Noun phrase structure | 40                  |
| Expressions of time   | 20                  |
| Verb forms            | 50                  |
| Pronouns              | 50                  |
| Sentence structures   | 65                  |

TABLE 1 – Division of the corpus.
Building the UNLization (analysis) Grammar

UNLization is the process of representing the content of a natural language structure using UNL. In order to UNLize any Natural language text, the UNLization (analysis) grammar for that natural language should be, first, developed. The UNLization reference grammars for English and Arabic reference corpora have been already built to represent the content of both corpora. English and Arabic grammars have common modules such as; the tokenization, numeral, attribute, syntactic and syntax-semantic modules; however, the Arabic analysis grammar has an extra module; namely, the transliteration module. The following sub-sections will describe each module.

2.1 The Tokenization module

The tokenization module is responsible for performing two tasks: The first is segmenting the input sentence into nodes (the tokens or processing units in the UNL framework). The second is preventing wrong lexical choices. The following sub-sections will examine each task.

2.1.1 Input segmentation

The tokenization algorithm is strictly dictionary-based; the system tries to match the strings of the natural language input against the entries existing in the dictionary. In case it does not succeed, the string is considered a temporary entry. There are no predefined tokens: spaces and punctuation marks have to be inserted in the dictionary in order to be treated as non-temporary entries. For instance, if the dictionary is empty, the string "Barking dogs seldom bite" will be considered a single token. If the dictionary contains only the entry [d], the input will be tokenized as [Barking][d][ogs sel][d][om bite]. The tokenization algorithm goes from left to right trying to match the longest possible string with dictionary entries, and it assigns the feature TEMP (temporary) to strings that are not found in the dictionary. “Temp” entries are written in natural language and are put between quotes in case of English; on the other hand, Arabic Temp nodes are transliterated into Latin characters first using the transliteration module that will be discussed in section 2.2. For instance, any URL such as "www.undlfoundation.org" should be considered TEMP; however, it is tokenized according to the entries found in the dictionary as [www] [.] [u] [nd] [l] [foundation] [.] [or] [g], which is incorrect since we expect the whole string to be treated as a single temporary entry. In order to avoid that, a disambiguation rule applies to consider any string as a single node if followed by blank space or is at the end of a sentence followed by a full stop.

2.1.2 Wrong lexical choice prevention

The tokenization algorithm blocks the segmentation of tokens or sequences of tokens prohibited by disambiguation rules. Disambiguation rules are not only responsible for the segmentation of the input, but also responsible for choosing the most appropriate sense to the context. For instance, “you” have two different realizations in the dictionary; the singular second person pronoun and the plural second person pronoun. In the sentence “you love yourself”, disambiguation rules should prevent the choice of the plural pronoun, thus, causing the engine to choose the singular pronoun if the verb is followed by a singular personal possessive pronoun.
2.2 Transliteration Module

This module is developed especially for the Arabic language. Some rules have been developed to transliterate words that are not found in the Arabic Analysis dictionary such as the temporary universal words (TEMP) into Latin characters. TEMPs will be analysed letter by letter; each letter will be transliterated to its equivalent Latin character, then, the whole word will be put between double quotes in the UNL representation.

2.3 The Numerals Module

Numerals in UNL are temporary UWs and should be represented in UNL as Digits between quotes. There are two cases in Numerals; they may be present in the input as digits in which case the engine will consider them as TEMP automatically, or, they may be written in letters, in the latter case the numerals module is activated. In order to handle numerals in both English and Arabic, both a dictionary and analysis rules are required. The numerals module is part of both the English and Arabic grammars. There are 4 types of numerals to be covered in this module: cardinal, ordinal, partitive and multiplicative. We will examine cardinal numbers first as they constitute the base for other types of numerals. There are many subsets of cardinal numbers such as units, tens, hundreds, thousands, millions…etc. The first step towards analyzing them is compiling a small dictionary that will enable rules to convert numbers in both English and Arabic into digits. Some cardinal numbers will be inserted in the dictionary as is such as the numbers from one to nineteen. Other numbers will be inserted incomplete in the dictionary to be later completed by rules; for example, tens are inserted without their zeros, ―twenty‖ is inserted as ―2‖…etc. The second step is to develop the required rules; units and numbers from ten to eleven are retrieved from the dictionary without any modification by rules. Tens starting from number twenty which are incomplete in the dictionary have two possibilities in analysis: The first is adding tens to units. For instance in the case of “twenty one‖, “twenty” which is stored in the dictionary as “2’ and “one” which is stored as “1” will be joined by a rule and will be treated as a single number “21”. The second is not adding tens to units as in “twenty”, a zero will be added to “2” and joined together by a rule to become “20”. If the input number is bigger as in hundreds, thousands, millions ….etc., the number of the possibilities used to analyse the number will increase. The numerals module has the ability to convert numbers from units to billions into digits. All numerals are analysed using the same logic shown in the flowchart in figure 1.

![Flowchart of Numerals UNL-ization](image-url)
The analysis of partitive numerals depends on their existence in the dictionary. In ordinal and multiplicative numbers; after converting the number in letters into digital numbers, an attribute “@ordinal” will be assigned to the number. If the number is followed by the word “times” such as “four times”, the attribute “@times” will be assigned to “4” to be “4.@times”.

2.4 Attributes module

In UNL, attributes have been used to represent information conveyed by natural language grammatical categories (such as tense, mood, aspect, number, etc). The set of attributes, which is claimed to be universal, is defined in the UNL Specs (http://www.unlweb.net/wiki/Attributes). The attributes module can handle determiners, pronouns, prepositions and verb forms. It is responsible for substituting certain words or morphemes with attributes, as in the case of quantity quantifiers (“a lot of”, “several”, “few”, “all”, “any”… etc.) which will be deleted and substituted by the attributes “@multal, @paucal, @any, @all …..etc.” to be assigned to the following word. In UNL, pronouns are “empty concepts” represented semantically as “00”. The person, number and gender of the pronoun are described by UNL attributes; the identity of the pronoun is defined by the UNL attributes. For example, the pronoun “he” and “she” are represented in UNL as “00” and both are third person and singular pronouns (00.@3.@singular) but “he” is a masculine and “she” is a feminine pronoun and, hence, their UNL representation will be respectively “00.@3.@singular.@male” and 00.@3.@singular.@female.

2.5 Syntactic module

After assigning the necessary attributes, the syntactic module should start drawing the syntactic trees for noun phrases, verb phrases and sentence structures that are part of the corpus, according to the X-bar theory (http://www.unlweb.net/wiki/X-bar_theory). The syntactic modules for Arabic grammar and English grammar both follow the same methodology, thus, the following subsections will present and discuss only English examples since they will be easier to understand. The syntactic module is divided into two phases; the list-to-tree phase and the tree-to-tree phase.

2.5.1 The list-to-tree phase

In this phase, rules are used to parse the tokenized input sentences into a tree structure. It starts by composing small trees for the small phrases in the sentence and then combining these small trees together to form a bigger tree. List-to-tree rules are responsible for building the trees for language structures; ordering of rules is required; rules for building noun phrase trees should be followed by rules for building verb phrase trees. The sentence “he buys books about cars from Paris” after being processed by the attribute module will be “he.@3.@male buy.@present book.@pl about car.@pl from Paris”. This sentence has two noun phrases, rules will start to project all nouns and pronouns in the sentence to the intermediate constituent “NB” and link the NBs to form a higher constituent from right to left. “cars” and “Paris” will be firstly linked together under the intermediate constituent “NB” and projected to the maximal projection NP by linking it to an empty node which will act as the specifier. As shown in figure 2, and the preposition “from” between the two NBs will be deleted leaving the semantic feature “frm” to be assigned to “Paris”.

1 A decision to link «from Paris » to the nearest head « car » was taken to solve the structural ambiguity concerning the PP attachment « from Paris ».
This feature will help later in the process of linking them with a semantic relation as will be shown in sub-section 2.6. then the NB “book” and the NP “cars from Paris” combined in figure 2 will be linked together under the bigger intermediate constituent “NB” and the preposition “about” will be deleted leaving the semantic feature “cnt” assigned to the NP “cars from Paris”. Then, rules will project the NB “book about cars from Paris” to the maximal projection NP by linking it with an empty node which will act as the specifier, and the NB of the pronoun “he” will also be projected to an NP since a pronoun does not need a specifier, this is shown in figure 3.

After building the tree of the noun phrase “books about cars from Paris”, rules will start to form a bigger tree that contains this noun phrase and the verb “buy” by combining them under the intermediate constituent “VB” as shown in figure 4. The transitive verb “buy” is carrying the feature “TSTD” in the dictionary; this means that it needs two syntactic arguments: a subject which is the doer of the action, and a single direct object. These two arguments are mapped semantically to an agent and an object. The two arguments should be “NPs”, thus, the semantic feature “obj” will be assigned to the NP “books about cars from Paris”. The biggest tree will be formed by combining the “VB” with the NP “he” under the maximal projection “VP” as in figure 5 and the semantic feature “agt” will be assigned to the pronoun “he”.

2.5.2 The tree-to-tree phase

The constituents “NB, NB, NP, VB and VP” will be mapped onto their syntactic roles. The methodology used in this phase is to start decomposing constituents from the biggest to the smallest. Ordering of rules is required; rules for decomposing verb phrase trees should be followed by rules for decomposing noun phrase trees. VP is the last constituent to be composed,
hence, it will be the first one to be decomposed. A key assumption of X-bar theory is that branching is always binary, thus, the decomposition of any constituent will affect the tree. The constituent will be decomposed to a syntactic role between a node and the head of the adjacent constituent to make the binary. In the present example, the decomposition of “VP” will affect the tree; “VS” or a Verb Specifier relation will be constructed between the verb and the pronoun as shown in figure 6. The second decomposed branch will be the “VB” which will be the Verb Complement relation ”VC” that constructed between the verb and the head “book” of the NP “book about cars from Paris” as shown in figure 7.

After first decomposing the VP and the VB since they are the biggest constituents in the tree, decomposing the smaller NPS and NBs will start. Because of emptiness of the specifier of the noun phrase, the NP syntactic relations between the empty nodes and the NBs will be deleted. The bigger NB “book about cars from Paris” will be decomposed into “book” and the head “cars” of the smaller NB “cars from Paris”, and the syntactic relation “NA” or Noun Adjunct will be established between them as shown in figure 8. Finally, the smallest constituent in the tree will be decomposed into the two nouns “car” and “Paris”, and the syntactic relation “NA” will be constructed between them as shown in figure 9. The output of the tree-to-tree phase; the four syntactic relations: VS, VC, NA and NA will be the input of the tree to Network phase.

### 2.6 Syntax – semantic module

In this module, rules have been built to derive the semantic network from the syntactic graph. Order of rules in this module are not necessary since the semantic features which are assigned to nodes from the list-to-tree phase make the rules constrained enough to be carried out in their context. As in the present example, the output of the tree-to-tree phase will be the input of this module or the tree-to-network phase; the VC, VS, NA and NA will be mapped with their corresponding semantic relations: “obj”, “agt”, “cnt” and “frm”, respectively.
There are two “NA” syntactic relations in the tree, but they will not be mapped onto the same semantic relations. Semantic features here play an important role in distinguishing between them; firstly, the feature “cnt” was assigned to “cars” in the phrase “books about cars” when the preposition “about” was deleted giving the meaning of content; its deletion left the semantic feature on the following noun. Secondly, the feature “frm” was assigned to the other noun, “Paris”, in the phrase “cars from Paris” when the preposition “from” was deleted, giving the meaning of origin since it is followed by a place noun; its deletion left this semantic feature on the following noun. The syntactic relation "VS" in this example has only a single possibility to be converted into a semantic relation; namely, an agt relation to show that “he” is the doer of the action of “buying”. The VC may be mapped to a goal or an object semantic relation. Ditransitive verbs such as “give” have two objects and, thus, make two syntactic relations “VCs” which are semantically mapped with “object” and “goal” relations; however, this is not the present case. The verb "buy" is a transitive verb that has only one object, hence, the “obj” semantic feature has been assigned to the noun “book” to map the VC with an object relation rather than goal. This way, the UNL graph or the final semantic representation for the present example will be completed as shown in figure 10. This semantic representation can be NL-ized back into natural language using the NLization grammar.

3 Building the NLization (Generation) Grammar

This section is concerned with the NL-ization of the reference corpus from the interlingua representation (UNL) into both Arabic and English. To achieve this purpose, Arabic and English linguistic resources have been developed. The resources are Arabic and English specialized dictionaries, in addition to Arabic and English NL-ization grammars. The process of generation may be seen to some extent as a mirror image of the analysis process. Similar to the process of natural language analysis, generating well-formed sentences has to pass through a set of grammar modules which are the semantic-syntactic module, the syntactic module, the attributes module, the numerals module, and a transliteration module.
3.1 The Semantic-Syntactic Module (Network-to-Tree Phase)

This module is the first module in the NL-ization grammar and is responsible for mapping the semantic relations onto their syntactic equivalents. As an example; the semantic graph in figure 10 which represents a verbal phrase requires mapping rules to map the semantic relations agt, obj, cnt, and frm onto their counterpart syntactic relations; Verb specifier (VS), Verb Complement (VC), Noun Adjunct (NA) and another Noun Adjunct (NA). Moreover, in case of the semantic relations are “cnt” and “frm” and their counterpart syntactic relation is noun adjunct (NA), mapping rules should also take into consideration whether the noun adjunct requires a preposition or not. After applying the mapping rules the syntactic network in (1) will be generated.

(1)

\[
\begin{align*}
\text{VS(verb;specifier);} \\
\text{VC(verb;complement);} \\
\text{NA(noun;PC(preposition;adjunct));} \\
\text{NA(noun;PC(preposition;adjunct));}
\end{align*}
\]

3.2 The Syntactic Module

The syntactic module is the second module of the NL-ization grammar, it is responsible for transforming the deep syntactic structure generated from the semantic-syntactic module into a surface syntactic structure. The Syntactic module is divided into two phases; the tree-to-tree phase and the tree-to-list phase. The tree-to-tree phase is responsible for gathering individual syntactic relations such as those in (1) and forming higher constituents and the tree-to-list phase is responsible for linearizing the surface tree structure into a list structure, the following two subsections will explain these two phases in more detail.

3.2.1 The tree-to-tree phase

In the tree-to-tree phase, rules are responsible for building the surface syntactic structure of the sentence by building the intermediate constituents (XBs) which are combined to form the maximal projections (XPs) and combined finally to form the sentence structure. For example, the syntactic relations VS, VC, and the two NAs in (1) will be combined to form the maximal projection VP, the intermediate constituent VB, and maximal projection NP according to the schema of X-bar theory as shown in figures 11, 12, 13, and 14.
In figure 13, the preposition (P) “عن” was inserted in the tree as the adjunct of the noun “كتاب” in the current example needs a preposition which is predicted by means of the semantic – syntactic module. Similarly, the preposition “من” was inserted in figure 14. The NP constituent in figure 14 is combined with the NP constituent in figure 13 to constitute the complement of the main verb “اشترى” as shown in figure 15. The verb complement in figure 15 will in turn be combined with the verb “اشترى” to form the intermediate projection VB as shown in figure 16.

Finally, the resulting VB shown in figure 16 is combined with the specifier (spec or VS) shown in figure 11 to build the final maximal projection of the phrase VP as shown in figure 17.
3.2.2 The tree-to-list phase

In the tree-to-list phase, rules are responsible for transferring the surface syntactic structure into a list structure and also adding the required spaces. Thus, the syntactic tree in figure (17) will be transformed into the list shown in figure (18) then (19).

\[ \text{Pro.@3.@male} \]

![Figure 17 – the syntactic tree of the final VP](image)

\[ \text{هو انتهى كتاب عن سيارة من باريس} \]

![Figure 18 – The prefinal list structure](image)

3.3 The Attributes Module

This module is responsible for converting the attributes represented in the interlingua into the suitable natural language words or affixes. For example, pronouns are represented in UNL as "00" nodes along with some attributes to reflect their number, gender,...etc. Table 2 shows examples of the different attributes used to distinguish between the different kind of pronouns and their Arabic and English counterparts. According to table 2, the pronoun in the list structure shown in figure 18 will be replaced by “هو”\(^2\) as shown in figure 19.

\[ \text{هو انتهى كتاب عن سيارة من باريس} \]

![Figure 19 – The final list structure](image)

---

\(^2\) A decision was taken to generate an overt pronoun to make the structure more explicit.
Moreover, there are many types of attributes represented in the UNL framework; all are handled during the NLization process. For example, an attribute expressing definiteness such as “@def” will be realized as the prefix “ال” during the generation of Arabic, and an attribute expressing number such as “@pl” will be realized for example as a suffix.

3.4 The Numeral Module

This module is responsible for converting digital numbers onto their counterpart natural language string (Arabic or English). There are four types of numerals to be covered in the numerals module; cardinal numbers, ordinal numbers, partitive numbers and multiplicatives. For cardinal numbers, the basic conversion mechanism is converting individual digits from (0 to 9) directly onto the counterpart natural language string, and then converting multiple digits by combining the converted individual digits to form bigger numbers. For example, with two digit numbers such as tens “10, 20, 30, 40, ...etc”, the first digit is converted into the Arabic strings "١٠", "٢٠", "٣٠", "٤٠", ... respectively. Then, the second digit is converted into "٠" and lastly, the two digits are combined together; for instance the number “10” is converted into “عشرة” in Arabic and “ten” in English.

The numerals module handles ordinal numerals; numerals that describe position or order such as “first”, “second”...etc. For example, the representation (65.@ordinal) will be NL-ized onto “الخامس والستون” in Arabic and “sixty-fifth” in English. The numerals module can also generate multiplicatives which describe repetition such as “once”, “twice”...etc. The distinctive difference between multiplicative numerals and other types of numerals in UNL representation is the attribute @times. For example, the UNL representation (3.@times) will be NL-ized into “ثلاث مرات” in Arabic and “three times” in English. Also, partitive numbers are handled; for example converting the partitive number “3 3/4” into “ثالثة وثلاثة أرباع” in Arabic and “three and three quarters” in English.

| Pronouns       | Arabic Pronoun | English Pronoun |
|----------------|----------------|-----------------|
| 00.@1          | “أنا”           | “I”             |
| 00.@1.@pl      | “نحن”          | “we”            |
| 00.@2.@female  | “أنتي”          | “you”           |
| 00.@2.@male    | “أنت”           | “you”           |
| 00.@2.@pl      | “أنتم”          | “you”           |
| 00.@3.@male    | “هو”           | “he”            |
| 00.@3.@female  | “هي”           | “she”           |
| 00.@3.@pl      | “هم”           | “they”          |
| 00.@reflexive.@1 | “نفسني”    | “my self”       |
| 00.@reflexive.@2 | “نفسك”    | “yourself”      |
| 00.@reflexive.@3.@male | “نفسه” | “himself”      |
| 00.@reflexive.@3.@female | “نفسها” | “herself”   |

Table 2 – some extracted pronouns
3.5 The Transliteration Module

This module is built especially for the Arabic language to transliterate temporary UWs that are not found in the Arabic generative dictionary into Arabic letters. This module is activated in the listing phase of the generation process. In this module rules transliterate each letter in the word into its Arabic counterpart. For example, the letter “j” in the temporary UW “john” is transliterated into the Arabic letter “ج”, “o” into “و”, and finally the letter “n” is transliterated into “ن”. Each letter will be marked with a certain feature then all will be concatenated to form the final transliterated word which is “جون”. During the generation process especially in the transliteration module, the developer takes into consideration that not all the TEMPs can be transliterated; for example, transliteration rules do not apply to e-mail addresses and websites.

4 Evaluation

The output of the UNLization process for both Arabic and English languages has been evaluated based on a corpus that is annotated manually semantically in order to figure out the quality and accuracy of the automatically generated semantic networks which is supposed to convey exactly the intended meaning of the natural languages. As mentioned before, the data set contained 500 Arabic and English sentences. Also, the output of the NLization process has been evaluated based on a manually translated corpus. In the UNL System, the F-measure (or F1-score) is the measure of a grammar's accuracy. It considers both the precision and the recall of the grammar to compute the score, according to the formula: 

\[ \text{F-measure} = 2 \times \left( \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \right) \]

Precision is the number of correct results divided by the number of all returned results but recall is the number of correct results divided by the number of results that should have been returned.

Precision measurement of the UNL-ized Arabic sentences was 0.98 while recall measurement was 0.96. Precision measurement of the UNL-ized English sentences was 0.979 while recall measurement was 0.96. Also, the same measurement was applied to figure out the correctness of the automatically generated Arabic and English languages from the UNL-ized documents; the precision measurement of the generated Arabic was 0.989 while recall measurement was 0.96. The precision measurement for the generated English was 0.97 while the recall measurement was 0.96. Based on the reported values of precision and recall of analysis and generation, F-measure of English-UNL (English analysis) is 0.969, Arabic-UNL (Arabic analysis) is 0.974, UNL-English is (English generation) 0.964 and UNL-Arabic (Arabic generation) is 0.974. The values report a very high similarity between the actual output and the expected output.

5 Conclusion and Future Work

This paper presented a formalized reference grammar for analyzing and generating Arabic and English within the UNL framework. The design of the reference grammar depended on linguistic phenomenon common to all languages in order to support the idea of an Interlingua. The evaluation of the current state reflected very high accuracy which can: first, be the base of a more robust system of machine translation; second, a support for other languages in the UNL system.
in order to synchronize themselves by building parallel corpora and analysis and generation grammars. This would also constitute objective criteria to compare results.

UNL as an Interlingua is expected to be used in several different tasks such as text mining, multilingual document generation, summarization, text simplification, information retrieval and extraction, sentiment analysis etc. Furthermore, in UNL-based systems there is no need for the source language to be different from the target language: an English text may be represented in UNL in order to be generated, once again, in English, as a summarized, a simplified, a localized or a simply rephrased version of the original.

Future work will be mainly directed to the reference corpus. It is planned to increase the number of structures from 500 to 1000, 5 sentences at least for every structure. Therefore, the minimal number of sentences to be processed in the next stage is expected to be 5000. The selection of such sentences will depend on their frequency; the most frequent 1000 structures selected from a running-corpus of 100,000,000 words. Accordingly, a more robust system for machine translation can be built and released for testing.

References

Alansary, S. (2010). A Practical Application of the UNL+3 Program on the Arabic Language. In Proceedings of the 10th International Conference on Language Engineering, Cairo, Egypt.

Alansary, S. (2011). Interlingua-based Machine Translation Systems: UNL versus Other Interlinguas. In Proceedings of the 11th International Conference on Language Engineering, Cairo, Egypt.

Alansary, S., Nagi, M., Adly, N. (2011). Understanding Natural Language through the UNL Grammar Work-bench, Conference on Human Language Technology for Development (HLTD 2011), Bibliotheca Alexandrina, Alexandria, Egypt.

Alansary, S., Nagi, M., Adly, N. (2010). UNL+3: The Gateway to a Fully Operational UNL System, 10th International Conference on Language Engineering, Ain Shams University, Cairo, Egypt.

AlAnsary, S. (2011), Interlingua-based Machine Translation Systems: UNL versus Other Interlinguas. 11th International Conference on Language Engineering, Ain Shams University, Cairo, Egypt.

Alansary, S. (2012). A UNL-based approach for building an Arabic computational lexicon, the 8th international conference on informatics and system (infos2012), Cairo, Egypt.

Boitet C. (2002). A rationale for using UNL as an interlingua and more in various domains. Proc. LREC-02 First International Workshop on UNL, other Interlinguas, and their Applications, Las Palmas, 26-31/5/2002, ELRA/ELDA, J. Cardeñosa ed., pp. 23—26.

Dorr, Bonnie J. (1987). UNITRAN: An Interlingua Approach to Machine Translation. Proceedings of the 6th Conference of the American Association of Artificial Intelligence, Seattle, Washington.

Dorr, Bonnie J. (1990). “A cross-linguistic approach to translation”. Proceedings of 3rd International Conference on Theoretical and Methodological Issues in Machine Translation of
Natural Language, Linguistics Research Center, University of Texas, Texas.

Dorr, Bonnie J., Hovy, E., Levin, L. (2004). *Machine Translation: Interlingual Methods*, Encyclopedia of Language and Linguistics. 2nd ed., Brown, Keith (ed.).

Nyberg E.H., Mitamura T. (1992). *The KANT System: Fast, Accurate, High-Quality Translation in Practical Domains*, in Proceedings of the International Conference on Computation Linguistics, (COLING 1992), Nantes, France.

Nyberg E. H., Mitamura T., Carbonell J. (1997). *The KANT Machine Translation System: From R&D to Initial Deployment*, in Proceedings of LISA (The Library and Information Services in Astronomy) Workshop on Integrating Advanced Translation Technology, Hyatt Regency Crystal City, Washington D.C.

Martins, R., Avetisyan, V. (2009). *Generative and Enumerative Lexicons in the UNL Framework*, the seventh international conference on computer science and information technologies (CSIT 2009), 28 September – 2 October, 2009, Yerevan, Armenia.

Martins, R. (2012). *Le Petit Prince in UNL*, the 8th international conference on language resources and evaluation (LREC’12), 23-25 May 2012, Istanbul, Turkey.

Uchida H., Zhu M. (2005). *UNL2005 for Providing Knowledge Infrastructure*, in Proceedings of the Semantic Computing Workshop (SeC2005), Chiba, Japan, 2005

Uchida H., Zhu M., (1993). *Interlingua for Multilingual Machine Translation*, in Proceedings of the Machine Translation Summit IV, Kobe, Japan.

Uchida H. and M. Zhu (2005). *UNL2005 for Providing Knowledge Infrastructure*, in Proceedings of the Semantic Computing Workshop (SeC2005), Chiba, Japan.

Uchida H. (1996). *UNL: Universal Networking Language – An Electronic Language for Communication, Understanding, and Collaboration*, UNU/IAS/UNL Center, Tokyo, Japan.

Witkam T. (2006). *History and Heritage of the DLT (Distributed Language Translation) project*, Utrecht, The Netherlands: private publication.