XMODEL: An XML-based Morphological Analyzer for Arabic Language

Mourad Gridach mourad_i4@yahoo.fr
Faculty of Sciences of Fez, Mathematics and
Informatics Department Sidi Mohamed
Ben Abdellah University
Fez, 30000, Morocco

Noureddine Chenfour chenfour@yahoo.fr
Faculty of Sciences of Fez, Mathematics and
Informatics Department Sidi Mohamed
Ben Abdellah University
Fez, 30000, Morocco

Abstract

Morphological analysis is an essential stage in language engineering applications. For the Arabic language, this stage is not easy to develop because the Arabic language has some particularities such as the phenomena of agglutination and a lot of morphological ambiguity phenomenon. These reasons make the design of the morphological analyzer for Arabic somewhat difficult and require lots of other tools and treatments. The volume of the lexicon is another big problem of the morphological analysis of the Arabic Language which affects directly the process of the analyzing. In this paper we present a Morphological Analyzer for Modern Standard Arabic based on Arabic Morphological Automaton technique and using a new and innovative language (XMODEL) to represent the Arabic morphological knowledge in an optimal way. Both the Arabic Morphological Analyzer and Arabic Morphological Automaton are implemented in Java language and used XML technology. Buckwalter Arabic Morphological Analyzer and Xerox Arabic Finite State Morphology are two of the best known morphological analyzers for Modern Standard Arabic and they are also available and documented. Our Morphological Analyzer can be exploited by Natural Language Processing (NLP) applications such as machine translation, orthographical correction, information retrieval and both syntactic and semantic analyzers. At the end, an evaluation of Xerox and our system is done.

Key words: NLP, Morphology, Arabic Morphological Analyzer, Morphological Automaton, XMODEL language.

1. INTRODUCTION

Nowadays, Arabic language faces many challenges due to a lot of reasons such as the increase of the Arabic web sites, Arabic media, and Arabic companies around the world using the Arabic language, etc as [5]. For these reasons, a lot of research in the domain has been developed to satisfy the increasing demand of the applications using Arabic language. Arabic morphology is one of the essential needs in this domain and lots of morphological analyzers are available now, some of them have a commercial purpose and the others are available for research and evaluation as [4]. Buckwalter Arabic Morphological Analyzer and Xerox Arabic Finite State Morphology are two of the best known morphological analyzers for Modern Standard Arabic and they are also available and well documented.

The morphological analysis of Arabic is interested, as of other languages, in the structure of the word. But being given the wealth of the Arabic word's structure and the problem of agglutination, the operation becomes more complex than in the other languages as [13]. Another source of the difficulty is that sentences are longer and more complex compared to other languages. The average length of a sentence is 20 to 30 words, and it often exceeds 100 words. We also note that diacritics are particularities for our language and it is also considered as another source of difficulty for morphological analysis. For all these reasons seen so far, Arabic language is conceived as one of the languages that present a big problem in the morphological analysis and make this process very complicated.

In this article we'll be presenting our Arabic Morphological Analyzer based on morphological automaton developed using Java language. The use of this language has some advantages like openness, standardisation, flexibility, and reusability. To develop this morphological analyzer, we used the particularities of the Arabic language that is concretized on multilevel: verbs and nouns are also characterized by a specific representation named the matrix "root – scheme". This representation will help us construct a morphological automaton for the Arabic language. We note that we have used a new and innovative language (XMODEL) to represent the Arabic morphological knowledge. The use of this new language helps us to reduce the number of the entries in the lexicon. It also makes our system very flexible and one of the best existing morphological analyzers for the Arabic language.

The structure of the article is as follows. First, in this introduction we discuss some challenges of Arabic language and the importance of morphological analyzers in Natural Language Processing. After that, we present some morphological analyzers for Arabic language related to our work in the second section. Then in the third section, we explain our approach for the choice of the linguistic resource. In section four, we present our Arabic Morphological Analyzer. In section five, we evaluate our morphological analyzer. In section six, we discuss the obtained results. Finally, in the last section, we draw some conclusions and future works to be done.

2. WORKS IN THE DOMAIN

Morphological processing involves two different tasks according to the operation type: generation and analysis. In generation we produce correct forms using given morphemes, while in analysis we try to identify morphemes for a given word. A lot of research has been done in the development of morphological analyzers for Arabic; some of them are available for research and evaluation, while the rest have a commercial purpose.

2.1. Buckwalter Arabic Morphological Analyzer (2004)

This analyzer is considered as one of the most referenced in the literature, well documented and available for evaluation. It is also used by Linguistic Data Consortium (LDC) for POS tagging of Arabic texts, Penn Arabic Treebank, and the Prague Arabic Dependency Treebank. It can be freely download from this address: http://www.nongnu.org/aramorph/french/. It takes the stem as the base form and root information is provided. This analyzer contains over 77800 stem entries which represent 45000 lexical items. However, the number of lexical items and stems makes the lexicon voluminous and as result the process of analyzing an Arabic text becomes long.

2.2. Xerox Arabic Morphological Analysis and Generation

Xerox Arabic morphological Analyzer is well known in the literature and available for evaluation and well documented. This analyzer is constructed using Finite State Technology (FST). It adopts the root and pattern approach. Besides this, it includes 4930 roots and 400 patterns, effectively generating 90000 stems. The advantages of this analyzer are, on the one hand, the ability of a large coverage. On the other hand, it is based on rules and also provides an English glossary for each word. But the system fails because of some problems such as the overgeneration in word derivation, production words that do not exist in the traditional Arabic dictionaries and we can consider the volume of the lexicon as another disadvantage of this analyzer which could affect the process of analyzing.

2.3. Darwish's Sebawai Morphological Analyzer

Sebawai is an Arabic Morphological Analyzer developed by (Darwish, 2003) in one day. The author affirms that his morphological analyzer was built in less than 12 hours using about 200 lines of Perl code. The analyzer is based on automatically derived rules and statistics. The aim of this analyzer is the generation of the possible roots of any given Arabic word with a rate of success that reaches 84%. The advantage of Sebawai is the rapidity of the generation of roots. A disadvantage of this analyzer, however, is it's can't give information about the word analyzed which make it very limited for the most applications in the domain.

2.4. Attia's Morphological Analyzer

This morphological analyzer is built using Finite State Technology and it is suitable for both analysis and generation. It is based on contemporary data (a corpus of news articles of 4.5 million words), and takes the stem as the base form. It contains 9741 lemmas and 2826 multiword expressions. The advantage of this system is the treatment of multiword expressions (MWEs). The system can efficiently handle compound names of people, places, and organizations. A disadvantage of the system, however, is its limited coverage. It does not handle diacritized texts and targets a particular application (syntactic parser).

2.5. ElixirFM: an Arabic Morphological Analyzer by Otakar Smrz

ElixirFM is an online Arabic Morphological Analyzer for Modern Written Arabic developed by Otakar Smrz available for evaluation and well documented. This morphological analyzer is written in Haskell, while the interfaces in Perl. ElixirFM is inspired by the methodology of Functional Morphology (Forsberg and Ranta, 2004) and initially relied on the re-processed Buckwalter lexicon as [9]. It contains two main components: a multi- purpose programming library and a linguistically morphological lexicon as [14]. The advantage of this analyzer is that it gives to the user four different modes of operation (Resolve, Inflect, Derive and Lookup) for analyzing an Arabic word or text. But the system is limited coverage because it analyzes only words in the Modern Written Arabic.

3. LINGUISTIC RESOURCE REPRESENTATION: THE XMODEL LANGUAGE

So as to develop a morphological analyzer of the Arabic language, representing the morphological knowledge becomes very crucial. Besides this, it's viewed as one of the central problems of the automatic processing of the Arabic morphology.

According to some works, in order to represent the morphological knowledge of the Arabic language, they have chosen to use the database concept as a basic support to store the morphological information as [3], [9] and [12]. To seek any information, they make use of requests consulting. Unfortunately, this method remains very limited to this type of challenges. Consequently, they do not give good results.

A second method of representation, which is widely used, is offered by the artificial intelligence. Accordingly, a morphological analyzer serves as an intelligent system able to infer the morphological nature of the analysed sentence from a certain knowledge-base which constitutes of data and morphological rules as [15]. However, the artificial intelligence language is criticized for its being general and sequential search of information. The choice of the Lisp or Prolog language as a support of representing the morphological knowledge may not probably be the right option. This is due to the fact that the interpreter is not well adapted to this kind of problems.

We can also mention a third method known as the semantic networks. This method is also used in the artificial intelligence as a support of knowledge representation. The semantic networks, already developed in relation with psychology, correspond to a graphic representation composed of a set of items called nodes linked with arcs: the nodes stand for concepts, while the arcs stand for the relationship between the concepts.

To achieve a better representation of the morphological knowledge of Arabic, we conceived an innovative language adapted for this specific situation: it's the XMODEL language (XML-based MOrphological DEfinition Language). XMODEL is based on the XML language setting profits of its advantages and particularities. As a result, all morphological entries are gathered in an XMODEL files. Using the new language helps direct search for information and determinism. It also enables us to represent the whole components, properties and morphological rules with a very optimal way. To clarify the last point, we note that our morphological database contains 960 lexicon items (morphological components) and 455 morphological rules to be applied to these morphological components which present a remarkable reduction in the number of entries in the lexicon compared to the existing systems (Xerox and Buckwalter). This representation helps us achieve the following goals:

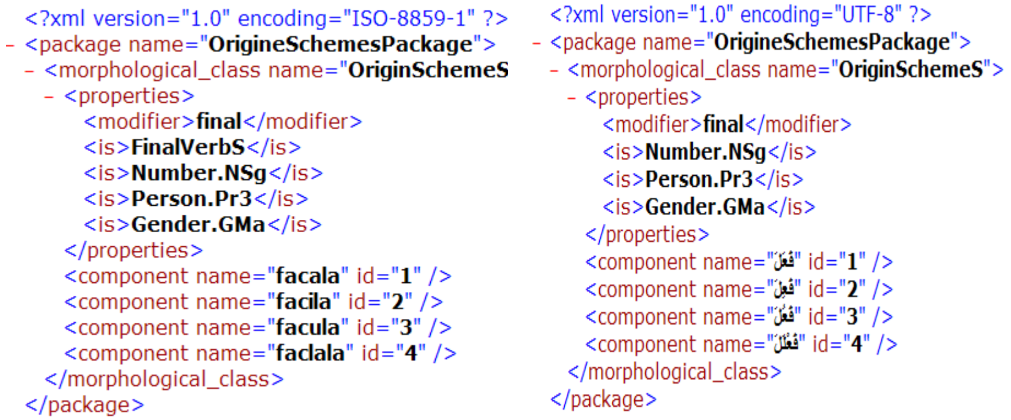
- ✓ A symbolic definition, declarative and therefore progressive of the Arabic morphology.
- ✓ A morphological database independent of processing that will be applied (see later).
- ✓ A considerable reduction of the number of morphological entries.
- ✓ The notion of scheme enables us to define the maximum morphological components by means of XMODEL language.

Our language makes it possible for us to represent the Arabic morphology as morphological classes and rules. Accordingly, our Arabic morphological database will be composed of three main parties: morphological classes, morphological properties and morphological rules.

Now let us first introduce the XMODEL language which permits to represent the morphological knowledge of Arabic and consists of three main parties. The first of which is:

3.1 Morphological components class

It enables us represent all morphological components of the Arabic language. It also permits to gather a set of morphological components having the same nature, the same morphological characteristics and the same semantic actions. Relying on the notion of scheme /*ealwazn*/ (الوزن), this class allows us a better optimization hence, a considerable reduction of morphological entries. By so doing, we needn't represent all the language items, but only their schemes. We note that our lexicon contains 960 items (morphological components) which is a remarkable reduction in the number of the items compared to the other dictionaries.



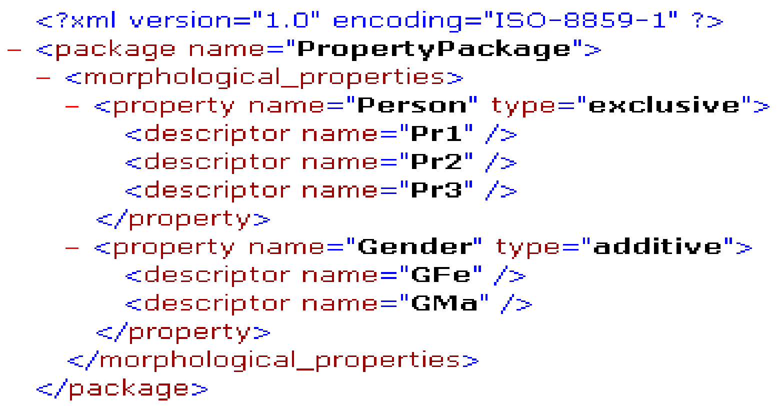
```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <package name="OrigineSchemesPackage">
  - <morphological_class name="OriginSchemeS
    - <properties>
        <modifier>final</modifier>
        <is>FinalVerbS</is>
        <is>Number.NSg</is>
        <is>Person.Pr3</is>
        <is>Gender.GMa</is>
      </properties>
      <component name="facala" id="1" />
      <component name="facila" id="2" />
      <component name="facula" id="3" />
      <component name="faclala" id="4" />
    </morphological_class>
  </package>
```

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <package name="OrigineSchemesPackage">
  - <morphological_class name="OriginSchemeS">
    - <properties>
        <modifier>final</modifier>
        <is>Number.NSg</is>
        <is>Person.Pr3</is>
        <is>Gender.GMa</is>
      </properties>
      <component name="فَعَلَ" id="1" />
      <component name="فَعِلَ" id="2" />
      <component name="فَعُلَ" id="3" />
      <component name="فَعْلَلَ" id="4" />
    </morphological_class>
  </package>
```

FIGURE 1: Representation of some verbs schemes using XMODEL language

3.2 Morphological properties class

It permits to characterize the different morphological components represented by the morphological class: a morphological property class contains a set of morphological descriptors or morphological values of properties that would be assigned to the different morphological components. We mention, for example, the property "*Gender*" which will distinguish between masculine and feminine components. The morphological properties are not related to a specific morphological class which makes it necessary to define them outside the morphological classes.



```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <package name="PropertyPackage">
  - <morphological_properties>
    - <property name="Person" type="exclusive">
        <descriptor name="Pr1" />
        <descriptor name="Pr2" />
        <descriptor name="Pr3" />
      </property>
    - <property name="Gender" type="additive">
        <descriptor name="GFe" />
        <descriptor name="GMa" />
      </property>
    </morphological_properties>
  </package>
```

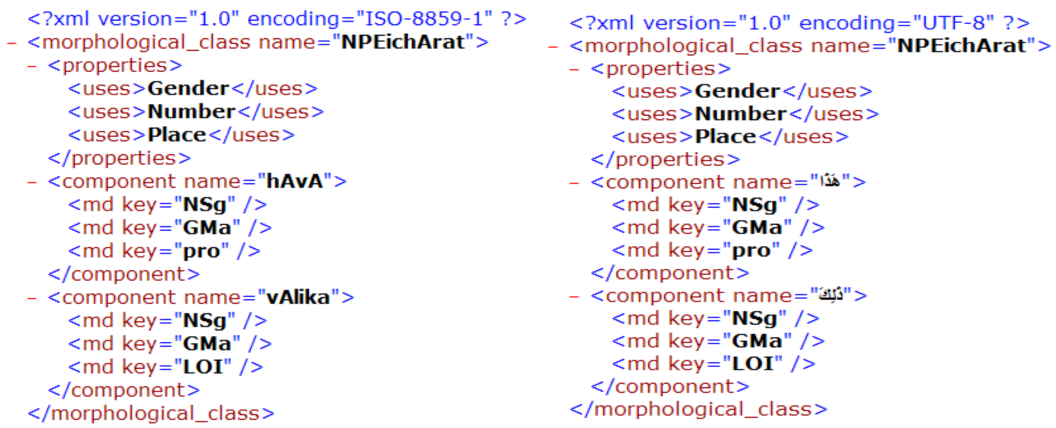
FIGURE 2: Representation of morphological properties "*Person*" and "*Gender*"

We have added the attribute "*type*" to work out the problem of the semantic of the morphological descriptors that might be **exclusive** (the morphological component can not be characterized by the morphological descriptors of the same property as in the case of the "*Person*" property) or **additive** (the morphological component can be characterized by the morphological descriptors of the same property as it is the case in the "*Gender*" property).

There are two strategies to characterize the morphological components using the properties:

3.2.1 Property of components

A morphological class can use a list of morphological descriptors to define its components generally speaking; each morphological component can have its own morphological descriptors. As for the "*Gender*" property, some components of this class can be masculine while the others can be feminine. This type of properties is named the **property of components**. In order to put them into practice, we have introduced the "*uses*" tag. This means that the different morphological descriptors defined by the property of components can be used by the different morphological components of the morphological class.



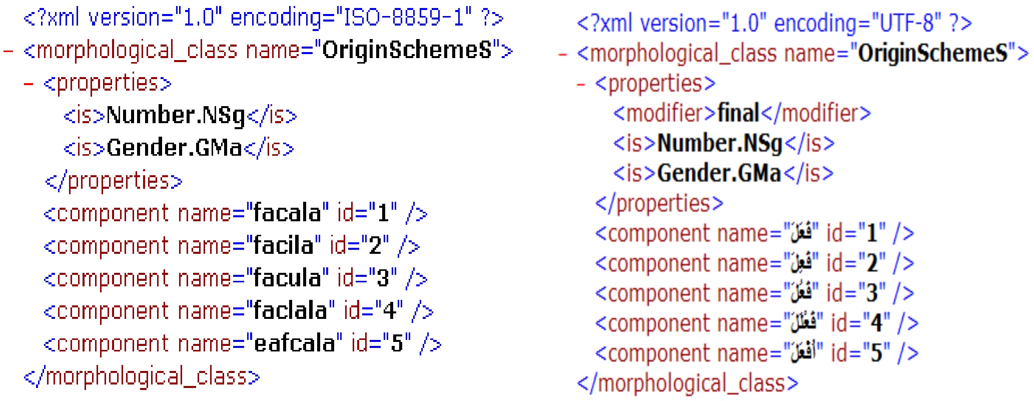
```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <morphological_class name="NPEichArat">
  - <properties>
      <uses>Gender</uses>
      <uses>Number</uses>
      <uses>Place</uses>
    </properties>
  - <component name="hAvA">
      <md key="NSg" />
      <md key="GMa" />
      <md key="pro" />
    </component>
  - <component name="vAlika">
      <md key="NSg" />
      <md key="GMa" />
      <md key="LOI" />
    </component>
  </morphological_class>
```

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <morphological_class name="NPEichArat">
  - <properties>
      <uses>Gender</uses>
      <uses>Number</uses>
      <uses>Place</uses>
    </properties>
  - <component name="هَذَا">
      <md key="NSg" />
      <md key="GMa" />
      <md key="pro" />
    </component>
  - <component name="ذَلِكَ">
      <md key="NSg" />
      <md key="GMa" />
      <md key="LOI" />
    </component>
  </morphological_class>
```

FIGURE 3: The property of components (« *Gender* » « *Number* » and « *Place* ») characterizing the components "*hAvA*" and "*vAlika*".

3.2.2 Property of classes

This one requires assigning a set of morphological components the commons morphological properties. For example, all components are masculine names. This type of property is known as **property of classes**. To realize this, we introduce the "*is*" tag.



```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <morphological_class name="OriginSchemeS">
  - <properties>
      <is>Number.NSg</is>
      <is>Gender.GMa</is>
    </properties>
    <component name="facala" id="1" />
    <component name="facila" id="2" />
    <component name="facula" id="3" />
    <component name="faclala" id="4" />
    <component name="eafcala" id="5" />
  </morphological_class>
```

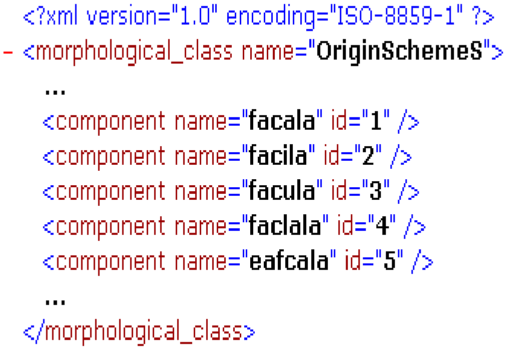
```
  <?xml version="1.0" encoding="UTF-8" ?>
- <morphological_class name="OriginSchemeS">
  - <properties>
      <modifier>final</modifier>
      <is>Number.NSg</is>
      <is>Gender.GMa</is>
    </properties>
    <component name="فَعَلَ" id="1" />
    <component name="فَعِلَ" id="2" />
    <component name="فَعُلَ" id="3" />
    <component name="فَعْلَلَ" id="4" />
    <component name="أَفْعَلَ" id="5" />
  </morphological_class>
```

FIGURE 4: Example of the property of classes

In the above example, all the schemes are singular components and masculine gender. It becomes evident to mention that the same class of the morphological components can use one combination of the tags "*uses*" and "*is*".

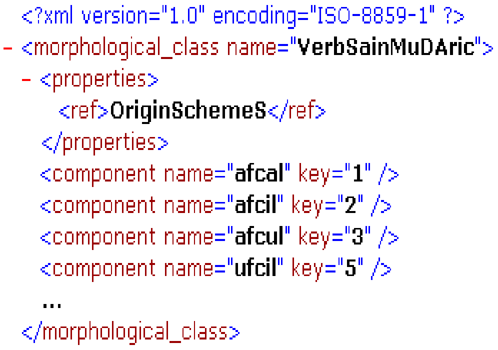
3.2.3 Property of reference

Another strong point of the XMODEL language is the introducing of the notion of **property of reference** which has an important role to benefit from the specificities of the Arabic morphology. As for the Arabic language some morphological components might be conjugated forms of other components which we call original components. An example of this is the case of the following components "*afcalu*", "*afcilu* ", "*afculu*". These components are all conjugated forms of the component "*facala*". We have specified this link of reference between the components using the "*ref*" tag.



```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <morphological_class name="OriginSchemeS">
    ...
    <component name="facala" id="1" />
    <component name="facila" id="2" />
    <component name="facula" id="3" />
    <component name="faclala" id="4" />
    <component name="eafcala" id="5" />
    ...
  </morphological_class>
```

FIGURE 5 : Example of some verbs schemes



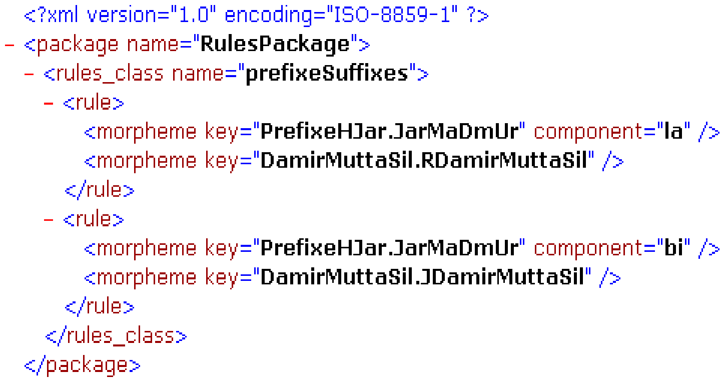
```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <morphological_class name="VerbSainMuDAric">
  - <properties>
      <ref>OriginSchemeS</ref>
    </properties>
    <component name="afcal" key="1" />
    <component name="afcil" key="2" />
    <component name="afcul" key="3" />
    <component name="ufcil" key="5" />
    ...
  </morphological_class>
```

FIGURE 6 : The conjugated forms of some verbs

In order to concretize this reference between the components, we have opted the attribute "*id*" to the original component. This attribute is specified in the "*component*" tag. The components that are conjugated forms will use this code as an attribute of that tag (the "*key*" attribute) to indicate this reference.

3.3 Morphological rules class

Firstly, it should be noted that we developed 455 morphological rules for the Arabic language. They help us combine some morphological components (morphemes) together to generate correct language words. They use the different morphological components classes as well as the morphological properties classes. The morphological rules classes allow us the possibility to add new morphological descriptors which do not belong to the union of morphological descriptors of components of rules. As a result, they are considered as a generator of language words. The implementation of the morphological rules class permits to put into practice all the possible concatenations between components.



```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <package name="RulesPackage">
  - <rules_class name="prefixeSuffixes">
    - <rule>
        <morpheme key="PrefixeHJar.JarMaDmUr" component="la" />
        <morpheme key="DamirMuttaSil.RDamirMuttaSil" />
      </rule>
    - <rule>
        <morpheme key="PrefixeHJar.JarMaDmUr" component="bi" />
        <morpheme key="DamirMuttaSil.JDamirMuttaSil" />
      </rule>
    </rules_class>
  </package>
```

FIGURE 7: Class of rules which represent the components prefixed by the prefix “*la*” and “*bi*”

In the above example, this rule permits to generate the components which begin by the prefix “*la*” and the prefix “*bi*”.

In a morphological rule, morphemes can have properties of different natures. Therefore, it is necessary to define with the rule of concatenation the strategy to manipulate the morphological descriptors of different morphemes. There are three possible strategies:

- *additive*: the gathering of the different morphological descriptors is assigned to the result word.
- *exclusive*: the concatenation doesn't take in any account the morphological properties of the components. Only the morphological descriptors of the rule are taken in account.
- *inclusive*: in this case, the same descriptors of the different morphemes are assigned to the concatenation.

The structuring of our morphological database using XMODEL language allows us to generate the morphological automaton of the Arabic language. In the next section, we will be dealing with the notion of morphological automaton.

4. SYSTEM DESCRIPTION

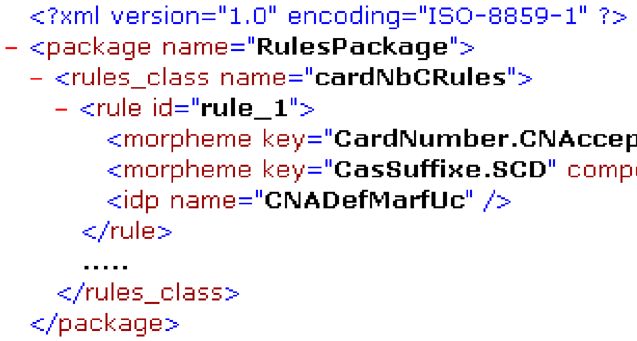
In this part, we describe the Arabic morphological analyzer. This latter is based on using morphological automaton technology. The implementation of each morphological analyzer for any language needs a main resource. This resource is the morphological database, so the first task is the conception and the realization of a morphological database. We also used a new language, XMODEL language, to create this database. After that, the second task is the development of a set of morphological automatons for Arabic language each of which represents a very definite category of morphology.

4.1. Arabic Morphological Automaton

To implement a morphological analyzer for any language, especially Arabic language, the use of morphological automaton is considered among the most efficient methods. It can be used for both analysis and generation. This latter is based on the notion of Finite State Automaton (DFA). A word is accepted by the morphological automaton if it belongs to a correct word in Arabic and rejected in the contrary case. Generally, the Arabic morphological automaton will have the following features:

- Q is a finite set of states of the control unit which represents the states of a morphological automaton.
- ∑ is a finite input tape alphabet symbols. Concerning morphological automaton for Arabic, it is constituted of the alphabets of Arabic language (Refer to figure1 in the Appendix to find all the Arabic alphabets).
- q_0 is the start state of the morphological automaton. It is constituted of only one start state in the case of a morphological automaton.
- F is a subset of Q. It also represents the accepting states of the morphological automaton for Arabic. This latter has a very important role because it permits to give us a set of information of the Arabic words analyzed. This information is called the morphological descriptors and they also characterize these words.
- The set τ also represents the transition function of the morphological automaton.

Consequently, the building of the morphological automaton of the Arabic language needs to use the XMODEL database discussed before. We have to extract all the morphological rules from this database and construct a morphological automaton of each rule. So to realize that constructing, we have to use some automaton operations such as concatenation and union operation. Let us clarify how we can use these two operations to generate a morphological automaton for a definite morphological rule. So, we consider the following rule:



```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <package name="RulesPackage">
  - <rules_class name="cardNbCRules">
    - <rule id="rule_1">
        <morpheme key="CardNumber.CNAccepteSCID.CNAccepteSC" />
        <morpheme key="CasSuffixe.SCD" component="u" />
        <idp name="CNADefMarfUc" />
      </rule>
      .....
    </rules_class>
  </package>
```

So to generate the morphological automaton which represent this rule, we have to use the operation of concatenation to concatenate the first morpheme (key = "*CardNumber.CNAccepteSCID.CNAccepteSC*") with the second one (key = "*CasSuffixe.SCD*" component = "*u*"). Therefore, the morphological automaton that represents this morphological rule is:

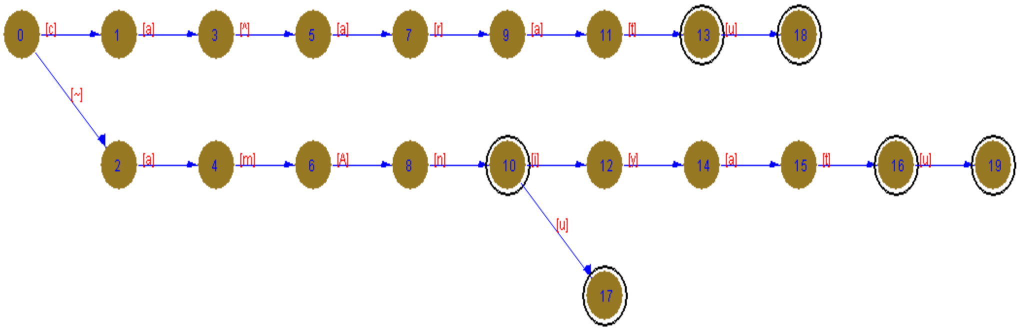
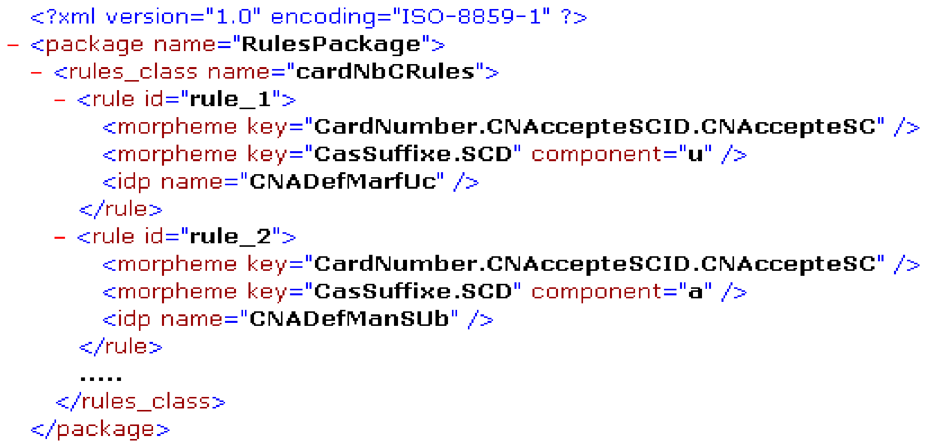


FIGURE 8: A morphological automaton representing the above morphological rule

In addition to the operation of concatenation used to concatenate morphemes or morphological automatons together, we used the union operation to associate two or several morphological automatons generated by the first operation, each one represent a definite morphological rule. To concretize the use of this second operation, let us consider the following class of morphological rules:



```
  <?xml version="1.0" encoding="ISO-8859-1" ?>
- <package name="RulesPackage">
  - <rules_class name="cardNbCRules">
    - <rule id="rule_1">
        <morpheme key="CardNumber.CNAccepteSCID.CNAccepteSC" />
        <morpheme key="CasSuffixe.SCD" component="u" />
        <idp name="CNADefMarfUc" />
      </rule>
    - <rule id="rule_2">
        <morpheme key="CardNumber.CNAccepteSCID.CNAccepteSC" />
        <morpheme key="CasSuffixe.SCD" component="a" />
        <idp name="CNADefManSUb" />
      </rule>
      .....
    </rules_class>
  </package>
```

In the above example, we have two morphological rules; each one generates a morphological automaton. We used the union operation to associate the first automaton which represents the rule identified by "*rule_1*" with the second automaton which represents the rule identified by "*rule_2*". The result morphological automaton is:

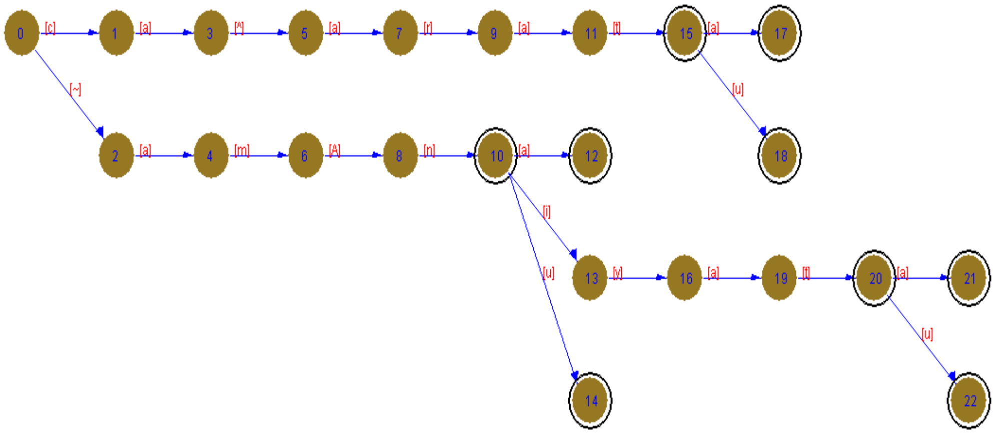


FIGURE 9: A morphological automaton representing the above rule

In the following paragraphs, we present a detail of how to construct all the morphological automatons of the Arabic language and the technical used in this constructing.

So as to build a morphological automaton of the Arabic language, we have classified words of the Arabic language in to two categories: the first category is that which submits to the derivation process, while the second one doesn't. This process of derivation is generated by a set of morphological rules known in the Arabic grammar under the name "*qawAcidu eaSSarfi*" /قواعد الصرف/. They repose on the manipulation of a set of very determined schemes named "*ealeawzAn*" /الأوزان/.

A scheme "*ealwazn*" /الوزن/ is an abstract linguistics term that represents a family of varied derived words; these words might be verbs or derived nouns which share the same linguistic features as [16]. At the graphical level, a scheme generally constitutes of:

- Three main consonants that are represented by the letters "f" /ف/, "c" /ع/ and "l" /ل/ with a possibility to duplicate the last letter "l" as in the case of schemes that correspond to a four letters root like "*faclala*" /فعلل/.
- Some consonants that serve as tools to extend the root like "*stafcala*" /استفعل/ and "*tafAcala*" /تفاعل/.
- A group of adequate vowels.

We have grouped in the first category of words the following items:

- Derived nouns "*ealaSmAe ealmu^taqqa*" /الأسماء المشتقة/.
- Strong verbs "*ealeafcAl eaSSaHiyHa*" /الأفعال الصحيحة/: these are the verbs that contain no weak letters. In the Arabic language, there are three weak letters: "w" /و/, "A" /ا/ and "y" /ي/.
- Weak verbs "*ealeafcAl ealmuctalla*" /الأفعال المعتلة/: these are the verbs that contain a weak letter. Weak verbs are also classified into three categories (ATTIA, 2005):

  a. Assimilated "*ealmi~al*" /المثال/: a verb that contains an initial weak letter.
  b. Hollow "*ealajwaf*" /الأجوف/: a verb that contains a middle weak letter.
  c. Defective "*eannAqiS*" /الناقص/: a verb that contains a final weak letter.

While the second category of words contains three families of words:

- The particular nouns "*ealasmAe ealxASSa*" /الأسماء الخاصة/: these nouns comprise proper nouns, names of cities, names of countries, etc. It also regroups the exclusive nouns "*easmAe ealeisti~nAe*" /أسماء الاستثناء/, the interrogative nouns "*easmAe ealeistifhAm*" /أسماء الاستفهام/, the demonstrative nouns "*easmAe ealei^Ara*" /أسماء الإشارة/, the conditional nouns "*easmAe ea^^art*" /أسماء الشرط/, etc.
- The particles "*ealHurUf*" /الحروف/ likes for example "*HurUfu ealjarri*" /حروف الجر/, "*HurUfu ealjazmi*" /حروف الجزم/, "*HurUfu ealcaTfi*" /حروف العطف/, etc.
- The incomplete verbs "*ealeafcAl eannAqiSa*" /الأفعال الناقصة/: this family of verbs contains the family of verb "*kAda*" /كاد/, the family of verb "*kAna*" /كان/ and the family of verb "*Zanna*" /ظن/.

Finally, after the constructing of our morphological automaton which contains the Arabic vocalized, its size is about 120 MB. We note that developing the morphological automatons of the Arabic language is the main idea of constructing a morphological analyzer for our language. So, in the following paragraph we will present our morphological analyzer for the Arabic language.

4.2. The Arabic Morphological Analyzer

First of all, because our morphological analyzer is based on the morphological automaton which is the main idea of this work, so this part it will be short than the part of the Arabic morphological automaton. So, in this section we describe our morphological analyzer for Arabic language. This analyzer is developed using three principal components:

- A morphological database constructed using the XMODEL language based on XML language integrating all the data suitable for Arabic language. Its regroups three packages: package of morphological components that contains verbs, nouns, particles and affixes. The second package includes the morphological rules and the last package is concerned with the morphological properties.
- A set of morphological automatons for the Arabic language each of which represents a very specific morphological category.
- A program handling the morphological database and the morphological automaton. It is developed through the use of Java language.

In addition, our morphological analyzer is meant to give a set of information about any Arabic word given to it. This set of information is about:

- The gender of the word: masculine or feminine.
- The person of the word: first, second or third person.
- The number of the word: singular, dual or plural.
- The case of the word: "*marfUc*" (مرفوع), "*manSUb*" (منصوب), "*majrUr*" (مجرور), "*majzUm*" (مجزوم).
- The type of the word: verb, noun or particle.

- If the word is a verb, we give its tense: present ("*ealmuDAric*": المضارع), past ("*ealmADI*": الماضي) or imperative ("*ealeamr*": الأمر). We also give its voice: active or passive.
- The origin scheme of the word is given if available.

We note that this set of information has an important role especially in future works like for example the building of a syntactic analyzer, a semantic analyzer, machine translation, etc.

Finally, the development of our morphological analyzer for Arabic language has many advantages such as:

- The separation between the task of the linguist and the developer.
- We can also reuse our programs in future works.
- Development standardization means in our application that we have build all the applications with the same standards, technologies (Java language, XML technologies, SAX, DOM, etc).
- Our morphological analyzer is developed using Java language. Therefore, our analyzer can be run in any platform such as Windows, Linux, UNIX and Mac OS.
- The facility of maintenance.

5. EVALUATION

In this section, we are going to evaluate Xerox Arabic Morphological Analysis and Generation and our Morphological Analyzer for the Arabic Language. We note that a standard annotated corpus for Arabic language is not yet available, for this reason the process of evaluation will be difficult. So, we have chosen Xerox Morphological Analysis and Generation because it is one of the best known morphological analyzers for Modern Standard Arabic and it is also available and well documented.

On the one hand, the first remark when we compare the two morphological analyzers is about the information giving by each one. Used an innovative language (XMODEL) for representing the morphological knowledge and the notion of the Morphological Automaton for Arabic Language, our morphological analyzer gives more information about each word analyzed and more precision compared to Xerox Arabic Morphological Analyzer. To clarify this point, let us take some Arabic words and try to analyze them using the two morphological analyzers:

The word	Morphological Analysis using Xerox Arabic Morphological Analyzer
صِفْرٌ [Sifrun]	CiCoC Noun +N Indef Nom
خَارِجُونَ [xArijUna]	CACiC participle Active +U3na Masc Plur Nom
مُرْتَدِّي [murtaddI]	muCtaCaC Participle Passive +I3 Ma Plur Acc/Gen Possessive
فُصِلْتُ [fuSiltu]	+tu 1stPer Masc/Fem Sing CuCiC Verb
أُخْرِجْتُمَا [euxrijtumA]	uCoCiC Verb +tumA 2ndPer Masc/Fem Dual
مَعَ [maca]	maEa Funcwa
أَمَامَ [eamAma]	CaCAC Noun +a Def Acc
العَاشِرَ [ealcA^ira]	al Article CACiC Noun +a Def Acc
بِهِمَا [bihimA]	bi +himA Funcwa
يُجَادِلُونَ [yujAdilUna]	yu Imperfect Prefix CACiC Verb +Una Indicative 3rdPer Masc Plur

TABLE 1: Words analyzed using Xerox Arabic Morphological Analyzer

The word	Morphological Analysis using our Arabic Morphological Analyzer
صِفْرٌ [Sifrun]	Gma Noun V0 Ind Raf [un]
خَارِجُونَ [xArijUna]	facala facila facula fAcil Gma NPl Noun efc Raf [Una]
مُرْتَدِّي [murtaddI]	eifcalla mufcallI Gma Pr1 NDl NSg Noun emf mmi8 KaS [I]
فُصِلْتُ [fuSiltu]	facula facala facila Gfe Gma Pr1 NSg Verb [tu]
أُخْرِجْتُمَا [euxrijtumA]	eafcala Gfe Gma Pr2 NDl Verb [tumA]
مَعَ [maca]	Particle zam mak Def NaS [a]
أَمَامَ [eamAma]	Particle mak Def NaS [a]
العَاشِرَ [ealcA^ira]	Noun V10 Def NaS [eal] [a]
بِهِمَا [bihimA]	Gfe Gma Pr3 NPl KaS [bi] [himA]
يُجَادِلُونَ [yujAdilUna]	fAcala Gma Pr3 NPl Verb Raf [yu] [Una]

TABLE 2: Words analyzed using our Morphological Analyzer

Related to the two tables above which represented the results of ten different Arabic words analyzed using the two morphological analyzers, we note that our morphological analyzer provides more information and more precision about the word analyzed compared to Xerox Morphological Analyzer. Thanks to our new innovative language (XMODEL) which permit to represent the morphological knowledge in an optimal way and the power of the morphological automaton for Arabic. This advantage will be very useful especially in the future works which will be done later. It should be noted that our system could provide more information about the word analyzed according to the user needs.

On the other hands, let us see the evaluation process from another view. So, we have selected a corpus of 975 words containing different type of the word in Arabic (verbs, nouns and particles). Then, we tested them on each morphological analyzer, and after that we draw a detailed analysis for the two analyzers. Our corpus contains 975 words divided into 481 nouns, 362 verbs and 132 particles. The table below shows the number of words which are not found when they are analyzed using the two morphological analyzers:

Type of the word	The number	Xerox Morphological Analyzer	Our System
Nouns	481	39	21
Verbs	362	16	-
Particles	132	29	-
Total	975	84	21

TABLE 3: Results of the evaluation process

To conclude this part of evaluation, using a new innovative language (XMODEL) and the notion of morphological automaton, our morphological analyzer can reach an average of performance around 95% which will make it one of the best existing morphological analyzers for the Arabic language and it will be very useful for the next future works to be done in NLP. We note that an update of our morphological database could resolve these errors seen in the table above. Represented as a set of XMODEL files, the process of updating the morphological database becomes very easy which make our innovative language one of the best languages to represent the Arabic morphological knowledge.

6. DISCUSSION

To compare our morphological analyzer for the Arabic language to the other existing systems, the task is difficult to do because there is no standard to make this comparison and every system has its own target. For this reason, each analyzer has some advantages and disadvantages comparing to the others.

Our morphological analyzer for the Arabic language has some advantages comparing to the others analyzers. These advantages are:

- Our morphological analyzer can be used in both analysis and generation
- It handles diacritized texts which permit to reduce the rate of ambiguity
- Our new and innovative language (XMODEL) used for the representation of the morphological knowledge and the use of morphological automaton for the Arabic Language permit to avoid a huge problems of ambiguity in the Arabic language which the most analyzers can't resolve
- The use of XMODEL language permit to reduce the number of the entries in the morphological database which present a big problem of the other morphological analyzers
- Represented as a set of XMODEL files, the process of updating the Arabic morphological database is very easy to develop. This advantage makes our system very flexible and one of the best existing morphological analyzers
- The major advantage of our system is that it permits, on the one hand, to give the affixes, the stem and the scheme (if the word is a noun or a verb and if it has a scheme) for any word given. On the other hands, it gives the information about the word analyzed using a list of morphological descriptors which permit to characterize every Arabic word

Our system has some disadvantages compared to some other systems. Firstly, it can't handle undiacritized texts. Secondly, it handles words which not exist in the Arabic language and finally, it doesn't provide an English glossary.

7. CONCLUSION & FUTURE WORKS

In this paper, we presented a Morphological Analyzer for Arabic language which is developed using a morphological database realized using XMODEL language and a set of morphological automatons for Arabic. The use of these automatons makes the system very efficient and fast. Another strong point of our morphological analyzer is the portability and the reusability because we have used Java for the development and the XML technology.

To extend our platform, we can also think to develop some works in the future:

- Develop the syntactic analyzer.
- Develop the semantic analyzer.
- Develop a system for Arabic learning.
- The help of the correction and the generation of texts.
- Automatic understanding of the texts: classification of texts, automatic summary and automatic extraction of the key words.
- Realize some specific applications such as machine translation, Q/R systems, Information Retrieval systems, etc.

Appendix

ا	:	A		س	:	s		ك	:	k
ب	:	B		ش	:	^		ل	:	l
ت	:	T		ص	:	S		م	:	m
ث	:	~		ض	:	D		ن	:	n
ج	:	J		ط	:	T		هـ	:	h
ح	:	H		ظ	:	Z		و	:	w
خ	:	X		ع	:	c		ي	:	y
د	:	D		غ	:	g		ى	:	A
ذ	:	V		فـ	:	f		ة	:	t
ر	:	r		ق	:	q		ء	:	e
ز	:	z								

Figure 1: Letter mappings

The short vowels	The long vowels
a : indicate the “*fatHa*”	A : represents the long vowel “ا”
u : indicate the “*Damma*”	U : represents the long vowel “و”
i : indicate the “*kasra*”	I : represents the long vowel “ي”

REFERENCES

1. Abouenour L., EL Hassani S., Yazidy T., Bouzouba K., Hamdani A. *“Building an Arabic Morphological Analyzer as part of an Open Arabic NLP Platform”*. In the Language Resources and Evaluation Conference (LREC), Marrakech, Morocco, 31st May, 2008

2. Attia, M. (2000). *“A large-scale computational processor of the Arabic Morphology and applications”*. Thesis submitted to the faculty of engineering, Cairo University

3. Attia M. (2005). *“Developing a Robust Arabic Morphological Transducer Using Finite State Technology”*. 8th Annual CLUK Research Colloquium. Manchester, UK

4. Attia, M. (2006). *“An Ambiguity-Controlled Morphological Analyzer for Modern Standard Arabic Modelling Finite State Networks”*. The Challenge of Arabic for NLP/MT Conference, the British Computer Society, London

5. Atwell E., Al-Sulaiti L., Al-Osaimi S., Abu Shawar B.. (2004). *“Un Examen d'Outils pour l'Analyse de Corpus Arabes”*. JEP-TALN 04, Arabic Language Processing, Fès, 19-22 April 2004

6. Beesley KR (1996). *“Arabic Finite-State Morphological Analysis and Generation”*. Proceedings of the 16th conference on Computational linguistics, Vol 1. Copenhagen, Denmark: Association for Computational Linguistics, pp 89-94

7. Beesley KR. 1998b. “*Arabic morphology using only finite-state operations*”. In Michael Rosner, editor, Computational Approaches to Semitic Languages: Proceedings of the Workshop, pages 50–57, Montreal, Quebec, August 16. Université de Montreal

8. Beesley KR (2000). *“Finite-State Non-Concatenative Morphotactics”*. SIGPHON-2000, Proceedings of the Fifth Workshop of the ACL Special Interest Group in Computational Phonology, p. 1-12, August 6, 2000, Luxembourg

9. Buckwalter T. (2002). *“Buckwalter Arabic Morphological Analyzer Version 1.0. Linguistic Data Consortium”*. University of Pennsylvania, LDC Catalog No.: LDC2002L49

10. Darwish K (2002). *“Building a Shallow Morphological Analyzer in One Day”*. Proceedings of the workshop on Computational Approaches to Semitic Languages in the 40th Annual Meeting of the Association for Computational Linguistics (ACL-02). Philadelphia, PA, USA

11. El-Sadany, T. A., Hashish, M. A. (1989). *“An Arabic Morphological System”*. IBM SYSTEM JOURNAL vol 28-no 4

12. Mars M., Belgacem M. (2006). *“Developed of a morphological analyser for Arabic language, tool for creation of educational activities of training of Arabic”*. Workshop "TEL in working context", 13-15 November 2006, Grenoble, France. 2006

13. Mars M., Belgacem M., Zrigui M., Antoniadis G., (2007). *“Analyseur morphologique de l'arabe”*. CITALA2007, 18-19 juin 2007, Rabat, Maroc

14. Otakar Smrz. ElixirFM. *"Implementation of Functional Arabic Morphology"*. In ACL 2007 Proceedings of the Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources, pages 1–8, Prague, Czech Republic, 2007.

15. Shaalan K. *"Extending Prolog for Better Natural Language Analysis"*. In Proceeding of the 1st Conference on Language Engineering, Egyptian Society of Language Engineering (ELSE), PP. 225-236, Egypt, March 14-15, 1998.

16. Tahir Y., Chenfour N., Harti M., *"Modélisation à objets d'une base de données morphologique pour la langue arabe"*. JEP-TALN 2004, Traitement Automatique de l'Arabe, Fès, 20 avril 2004