ORIGINAL METHOD FOR ROMANIAN TEXT-TO-SPEECH SYNTHESIS BASED ON SYLLABLE CONCATENATION

Ovidiu BUZA*, Gavril TODEREAN*, Alina NICA*, Zsolt BODO**

*Department of Telecommunications, Technical University of Cluj, Romania
**Siemens PSE Program and System Engineering Ltd., Budapest, Hungary

Corresponding author: Ovidiu Buza, 26 – 28 Baritiu Str., 400027, Cluj-Napoca, Romania
tel: 0744 367863, fax: 0264 591689, mail: Ovidiu.Buza@cs.utcluj.ro,
Toderean@pro3soft.ro, Alina.Nica@com.utcluj.ro, Zsolt.Bodo@siemens.com

We present in this article our method for text-to-speech synthesis based on syllable concatenation. The method consists of two main text processing stages. In the first stage input text is processed for detection of sentences, words, and syllables. Units are detected by a LEX-generated text parser. In the second stage syllables are marked as common or stressed syllables by mean of another LEX parser. For testing of our method a TTS prototype has been implemented. The prototype invokes the text parser, matches the units from an acoustical database and realizes speech synthesis.

Key words: Rule-driven text analysis, syllable and stress detection, text-to-speech synthesis.

1. INTRODUCTION

This article presents an original approach for constructing a syllable-based TTS system for Romanian. Syllable-base methods are an alternative to large corpora methods, as they give good results using a smaller database with acoustic units. Using of syllables in synthesis also leads to a good level of speech naturalness and low concatenation error rate because of small number of concatenation points inside the synthesized text.

The syllable approach is very appropriate in Romanian case, because Romanian spoken language contains a large number of opened vowels that gives a constant rhythm of speech and similar manner of accentuating words.

The particular aspects of our work are:

- using of linguistic and phonetic rules based of which we have done text analysis and obtained appropriate units and prosody data;
- constructing a vocal database with a subset of Romanian syllables;
- using of a specific unit matching procedure for retrieving data;
- using a concatenation algorithm for generating chain units.

These aspects were implemented into a TTS system prototype, as one can see in following diagram:

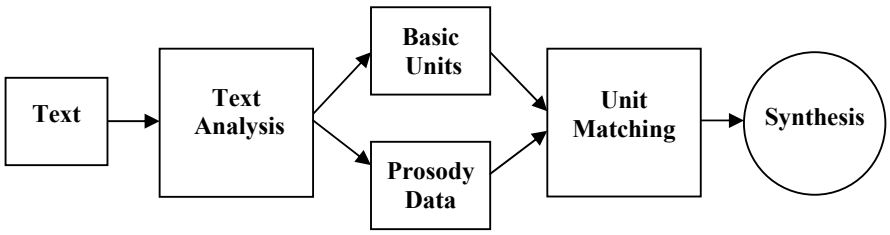

Figure 1. Main modules of our Text-to-Speech prototype.

The TTS prototype contains following modules:
– a text analysis module that brings input text and produces basic units, that in our approach are syllables, and prosody data, which mean the information about how words are accentuated;
– a unit matching module that generates acoustic units sequence according to the linguistic units detected from the input and prosody data;
– a speech synthesis module that generates speech based on the acoustic units sequences.

2. TEXT ANALYSIS

First stage in text analysis is the detection of linguistic units: sentences, words and segmental units that in our approach are the word syllables.

Detection of sentences and words is done based on punctuation and literal separators. For detection of syllables we had to design a set of linguistic rules for splitting words into syllables, inspired from Romanian syntax rules ([2], [3]).

The principle used in detecting linguistic units is illustrated in figure 2. Here we can see the structure of text analyser that corresponds to four modules designed for detection of units, prosody information and unit processing.

These modules are:
– a lexical analysis module for detection of basic units;
– a phonetic analysis module for generating prosody information;
– a high level analysis module for detection of high-level units;
– the processing shell for unit processing.

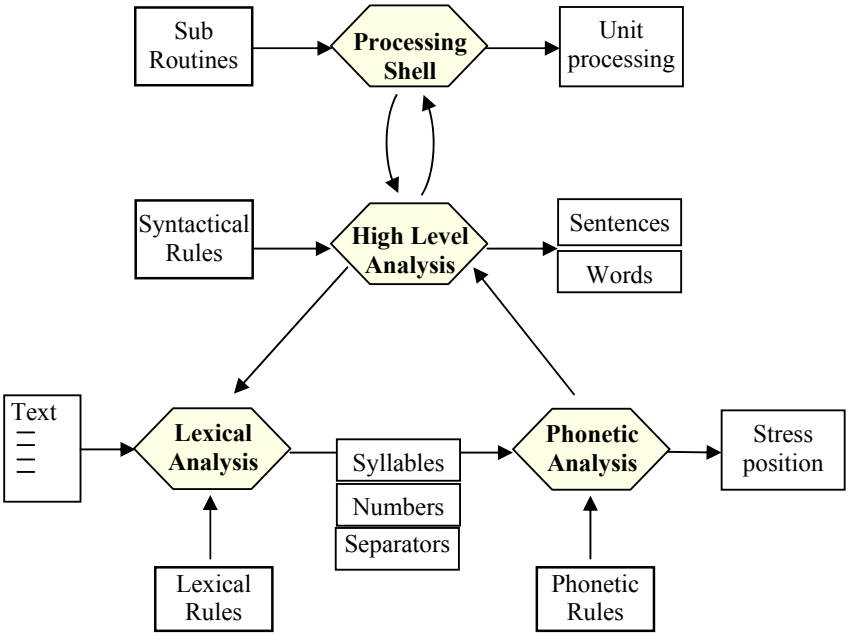

Figure 2. Text analyser for syllable detection.

Processing shell accomplishes the unit-processing task and controls the subsequent modules. The shell calls high-level analyser for returning main syntactic units. High-level analyser calls the lexical analyser for input text parsing and detection of basic units. Then phonetic analysis module is called for generating stress information.

Lexical analyser extracts text characters and clusters them into basic units. We refer to the detection of alphabetical characters, numerical characters, special characters and punctuation marks. Using special lexical rules (that have been presented in [8]), alphabetical characters are clustered as syllables, digits are clustered as numbers and special characters and punctuation marks are used in determining of word and sentence boundaries.

Phonetic analyser gets the syllables between two breaking characters and detects stress position, i.e. the accentuated syllable from corresponding word.

Then, high-level analyser takes the syllables, special characters and numbers provided by the lexical analyser, and also prosodic information, and constructs high-level units: words and sentences. Also basic sentence verification is done here.

Processing shell finally takes linguistic units provided from the previous levels and, based on some computing subroutines, classifies and stores them in appropriate structures. From these structures, synthesis module will construct the acoustic waves and will synthesize the text.

3. LEXICAL ANALYSIS FOR SYLLABLE DETECTION

Lexical analyser is called by the higher level modules for detection of basic lexical units: syllables, breaking characters and numbers. The lexical analyser is made by using LEX scanner generator [4]. LEX generates a lexical scanner starting from an input grammar that describes the parsing rules. Grammar is written in BNF standard form and specifies character sequences that can be recognized from the input. These sequences refer to syllables, special characters, separators and numbers. Also BNF grammar specifies the actions to be taken in the response of input matching, actions that will be accomplished by the processing shell subroutines.

The whole process realized by the lexical analyser is illustrated in figure 3. As we can see, input text is interpreted as a character string. At the beginning, current character is classified in following categories: digit, special character or separator, and alphanumeric character. Taking into account left and right context, current character and the characters already parsed are grouped to form a lexical unit: a syllable, a number or a separator. Specific production rules for each category indicate the mode each lexical unit is formed and classified, and also realize a sub-classification of units (for numbers if they are integer or real numbers, and for separators – the type: word or sentence separator, affirmative, interrogative, imperative or special separator).

Once the unit type and subtype is identified, corresponding character sequence is stored and transmitted to the high-level analyser by mean of specific actions, as they will be described in next paragraph (***Proceed syllable, Proceed number, Proceed separator***).

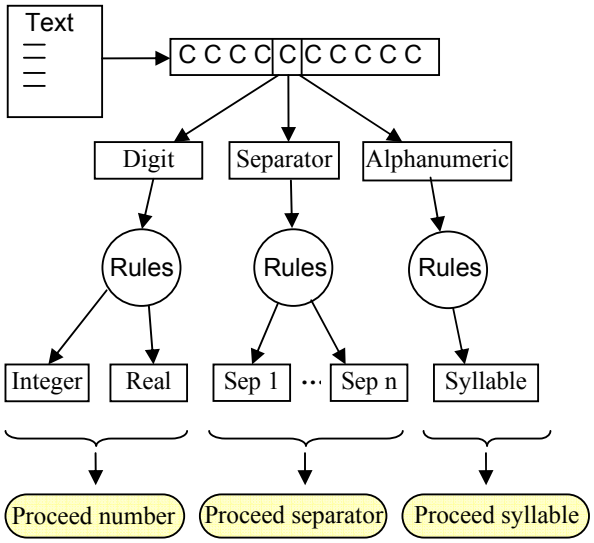

Figure 3. Lexical analyser for syllable detection.

3.1. Specific actions in lexical analyser

Specific actions inform high-level module about matching of syllables, numbers and breaking characters. Inside lexical parser three types of input response actions are defined as follows:

1. Proceed syllable – this is the action to be taken when a syllable is matched in specific location of one word.

Special attention is taken when a syllable is matched at the beginning of a word. In Romanian, different word decomposition rules apply when a character sequence occurs at the beginning or in the middle or the final part of a word.

2. Proceed number – is the action to be taken when a number is matched from the input. The number is identified as INTEGER or REAL type. In future stage, numbers will be translated in orthographic alphabetical form.

3. Proceed separator – is the action corresponding to a breaking character matching from the input. Breaking characters and punctuation marks are used for detecting word and sentence boundaries.

3.2. Syllable rules matching

Regarding rule matching process inside lexical analyser, two types of rule sets were made: a basic set consisting of three general rules, and a large set of exception rules which states the exceptions from the basic set.

(A) The basic set shows the general decomposition rules for Romanian.

First rule is that a syllable consists of a sequence of consonants followed by a vowel:

`syllable = {CONS}*{VOC}` (R1)

Second rule statues that a syllable can be finished by a consonant if the beginning of the next syllable is also a consonant:

`syllable= {CONS}*{VOC} {CONS}/ {CONS}` (R2)

Third rule says that one or more consonants can be placed at the final part of a syllable if this is the last syllable of a word:

`syllable= {CONS}*{VOC} {CONS}*/ {SEP}` (R3)

(B) The exception set is made up from the rules that are exceptions from the three rules of above. These exceptions are situated in the front of basic rules. If no rule from the exception set is matched, then the syllable is treated by the basic rules. At this time, the exception set is made up by more then 100 rules. Rules are grouped in subsets that refer to resembling character sequences. All these rules were explained in [7]. In this paper only few examples are shown at each group of rules (rules are grouped in subsets that refer to resembling character sequences).

First group of rules considers two-vowel sequences situated at the final of the word (‘`ei`’ ‘`ua`’, ‘`ou`’, ‘`eu`’, ‘`oi`’ sequences have been considered). For example words like *macrou* have to be decomposed in ‘`ma-crou`’ as following:

`{CONS}*ou/ {SEP}` {**Proceed**_`Syllable;}`

Next rules consider word-ending sequences finished by a soften ‘`i`’ vowel (like in many plurals of Romanian words or some verb terminations). Here plurals like « `{VOC} RI`» have been considered (`{VOC}` means a vowel), like in *gru-puri*, *co-cori* . Other examples of plurals traced here are: *sub-tili, măşti, terţi puşti, caşti, stu-denţi, urşi, stinşi*. Present tense of second singular person verbs is also considered, like *ci-teşti, gă-seşti*.

An example of rule from this set is:

`{CONS}*e [cnsş]*[tţ] i/ {SEP}`

```
/* is-teți, stu-denți, repe-tenți*/
{ Proceed_Syllable;}
```

Special cases refer to groups of two adjacent vowels. In some cases these groups act like diphthongs (both vowels being part of the same syllable), but in other cases they don't and the two-vowel group is divided in two distinct syllables (we say they act like hiatus).

For example <IA> group acts like a diphthong in *bă-iat* and like a hiatus in *Ma-ri-a*, *A-dri-a-na*. After main set rules, <IA> is not a diphthong and in main cases will be automated separated in distinct syllables. So in the exception set we have to indicate the cases when the vowels act together as a diphthong, like in following rule:

```
ia {CONS}/ {CONS}     |
ia {CONS}/ {SEP}{ Proceed_Syllable;}
/* iar-na, bă-iat*/
```

Another case is that of CI, CE, CHI, CHE, GI, GE, GHI, GHE groups. 'C' and 'G' followed by an 'I' or 'E' have specific behaviour and they will be always placed in same syllable. If the groups <[CG]I> or <[CG]HI> are followed by the vowel 'U' then 'IU' acts like a diphthong:

```
{CONS}*{VOC}/[cg][h] /* ma-CHe-ta */
                  { Proceed_Syllable;}

[G]IU/ {SEP} | [G] I/UL {SEP}  |
[C]H*{IU} | [C] H*{IU} {CONS} +/ {SEP}

{ /* ve-CHIUL;  barcaGI-ul*/
     Proceed_Syllable;}
```

Next 'U{CONS}I', 'U{CONS}{CONS}I' word-ending groups are considered. These groups will be placed in the same syllable as follows:

```
{CONS}?{CONS}?uni/{SEP} { /* co-muni, pruni */
        Proceed_Syllable;}

{CONS}*u[ln][șț]i/{SEP}    { /* că-runți, des-culți */
        Proceed_Syllable;}
```

Also IU and IE groups act differently, some cases as diphthongs, some cases as hiatus. Diphthong cases are illustrated in the rules:

```
{CONS}{IU} |({IU}|IE) |
({IU}|IE){CONS}/{CONS}
{ /* iu-bi-re; iur-ta ; ie-le; ier-na*/
        Proceed_Syllable;}
```

Other groups of vowels have been considered because they have not regular behaviour. IO and EA groups also act like diphthongs ('*cio-ban*', '*tea-tru*') and as hiatus ('*pi-o-let*', '*re-al*'). In our approach only OA,UĂ,ÎI,ÂI groups have been considered as diphthongs in all situations.

4. SYLLABLE ACCENTUATION

The principle for determining syllable accentuation is shown in Figure 4.

The parser returns current word from input stream. The word consists of series of phonemes F_1, F_2, ..., F_k and is delimited by a separator *S*. Phonetic analyser reads this word and detects syllable accentuation based on phonetic rules.

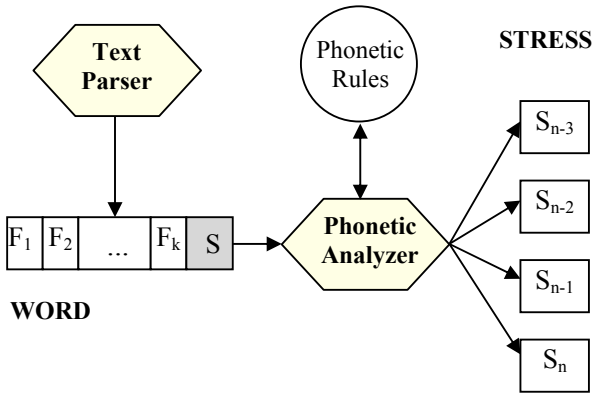

Figure 4. The principle of detecting syllable accentuation.

In Romanian, stressed syllable can be one of last four syllables of the word: S_n, S_{n-1}, S_{n-2} or S_{n-3}, (S_n is the last syllable). Most often, stress is placed at next to last position.

The rule set consists of:

(a) **one general rule**: S_{n-1} syllable is stressed:

```
{LIT}+/{SEP}     { return(SN_1);}
```

and

(b) a consistent **set of exceptions**, organized in classes of words having the same termination.

Unlike syllable detection analyser, in stress detection case we don't have specific actions for each rule (like *Proceed_number, Proceed_syllable*). Here only index of stressed syllable is returned.

First case in exception set consists from Romanian verbs ended with `'A'` vowel (infinitives or past tense). At this moment, we have inserted a limited set of these verbs, but in the final version we shall use all the verbs from a Romanian dictionary. In this case, last syllable (S_n) will be stressed. From example:

```
(picur|lu|calcul|furiş|hotăr|judec)a/{SEP}  |
(mir|scri|pict|desen|juc|disloc|(de)?conect)a/{SEP} |
(stud|răsfo|cop|tă|chinu|răsfo|preţu|miru) ia/ {SEP}
                {return(SN) ;}
```

Also, some noun plurals are exceptions from basic rule. These are *II* ending words, and also articulate or non-articulate nouns ended with *LE*:

```
(scat|geamg|coliv|cop|pust|pălăr|cut|terap)ii/{SEP}
                      /* scatii, copii*/
                      {return(SN) ;}

(bre|a|cas|s|be|sal)tele/{SEP}   {/* bretele, atele, castele */
                                 return (SN_1);
                                 }

{LIT}+(d|l|m|n|t)ele/{SEP} {/* peretele,stelele,numele,colindele*/
                                 return (SN_2);
                                 }

{LIT}+g[eu]rele/{SEP}       { /* fagurele,strugurele */
                            return (SN_3);
                            }
```

Then *EA*-ending words (articulate nouns or past tense verbs) follow:

```
{LIT}+(e|a)rea/{SEP}  {/* mâncarea, studierea, puterea */
                                return (SN_1);
                                }

{LIT}+[nr]tea/{SEP}   {/* cartea, mintea, puntea */
                                return (SN_1);
                                }

(origi|margi|nime|aseme|creme|funingi) nea? / {SEP}    {
                                /* marginea */
                                return (SN_2);
                                }

{LIT}+ea/ {SEP}       {/* şosea, andrea; putea, plăcea */
                                return (SN);
                                }
```

Other exceptions are RI and RII ending nouns. These are plural nouns, articulated or not:

```
{LIT}+[bdlmnrt]ări/{SEP} {/* mutări */
                                return (SN);
                                }

(nastu|marto)rii/{SEP} {/* nasturii, martorii  */
                                return (SN_2);
                                }


(na|sta|ma|fac|cap|rup|pic|mon|tor|struc|f[ală]p)turi/ {SEP} {
                                /*  fac-turi, na-turi*/
                                return (SN);
                                }

(mărtu|dato|căsăto|pală|bucu)rii/{SEP} {
                                /* mărturii, pălării */
                                return (SN);
                                }

{LIT}+rii/{SEP}       {/*  senatorii  */
                                return (SN_1);
                                }
```

Then follows a category of word ending with *T, TI, Ţ, ŢI, S, ŞI.* These words can be nouns, adjectives, verbs at present or past tense:

```
{LIT}+eşi/ {SEP}{/* răzeşi culeşi */
                                return (SN);
                                }

{LIT}+a[cr][tţ]i?/{SEP}         {/* compact(ţi), imparţi */
                                return (SN);
                                }

{LIT}+u[ln][şţ]i/{SEP}          {/* ajunşi, cărunţi, desculţi */
```

```
                                return (SN);
                                }

{LIT}+i[sş][tţ]i?/{SEP}         {/* poliţist, navetist */
                                return (SN);
                                }
```

Then words with *I, O, U* vowels in last syllable were considered:

```
{LIT}+i[lnsştţv]i? / {SEP} {/* fi-til, fu-riş, ciu-lit, pi-tit  */
                                return(SN);
                                }

{LIT}+{VOC}u/ {SEP}        {/*  tablou, maieu, saleu */
                                return (SN);
                                }

(pic|can|mar|vec|Nis)tori? / {SEP} {/* pictor, cantor, vector */
                                      return (SN_1);
                                      }

{LIT}+u(r|[ştţ]i?|ni?)/{SEP}  {/* plăcut, acţiuni, matur */
                                return (SN);
                              }
```

Other category is represented by *RA, RE, REA, RĂ, RI* ending words:

```
{LIT}+[jm]ur[ăe]/ {SEP}    {/*  tremură, latură */
                                return (SN_2);
                                }

(fal|co|mo|car)duri/{SEP}  {/*  fal-duri*/
                                return (SN_1);
                                }

{LIT}+[dirt]eri/{SEP} {/* că-deri, pă-reri, pu-teri */
                                return (SN);
                                }
```

We must say that only few examples of rules have been exposed here, but in [7] one can find the complete set.

5. VOCAL DATABASE AND UNIT MATCHING

Vocal database includes a subset of Romanian language syllables. Acoustic units were separated from male speech and normalized in pitch and amplitude. Segmentation was done through a semi-automated procedure that can detect silence/speech and voiced/unvoiced signal.

Vocal database with recorded syllables has a tree data structure. Each node in the tree corresponds with a syllable characteristic, and a leaf represents appropriate syllable.

Units have been inserted in database following the classification:

– after length of syllables: we have two, three or four character syllables (denoted S2, S3 and S4) and also singular phonemes (S1);
– after position inside the word: initial or median (M) and final syllables (F);
– after accentuation: stressed or accentuated (A) or normal (N) syllables.

This classification offers the advantage of reducing time for matching process between phonetic and acoustic units.

The matching process is done according to the three–layer classification of units: number of characters in the syllable, accentuation and the place of syllable inside the word.

If one syllable is not founded in vocal database, this will be constructed from other syllables and separate phonemes that are also recorded. Following situations may appear:

(a) Syllable is matched in appropriate accentuated form. In this case acoustic unit will be directly used for concatenation.

(b) Syllable is matched but not the accentuation. In this case, unit is reconstructed from other syllables and phonemes that abide by the necessary accentuation.

(c) Syllable is not matched at all, so it will be constructed from separate phonemes.

6. IMPLEMENTATION

The final aim of our work was to build a speech synthesis system based on concatenation of syllables. The system includes a syllable database in which we have recorded 386 two-character syllables: 283 middle-word syllables and 103 ending-word syllables, 139 most frequent three-character syllables and 37 four-character syllables. Syllables that are not included in database are synthesized from existing syllables and separate phonemes that are also recorded.

The speech synthesis system first invokes text analyser for syllable detection, then phonetic analyser for determining the accentuation. Appropriate unit (stressed or unstressed) is matched from vocal database, and speech synthesis is accomplished by syllable concatenation.

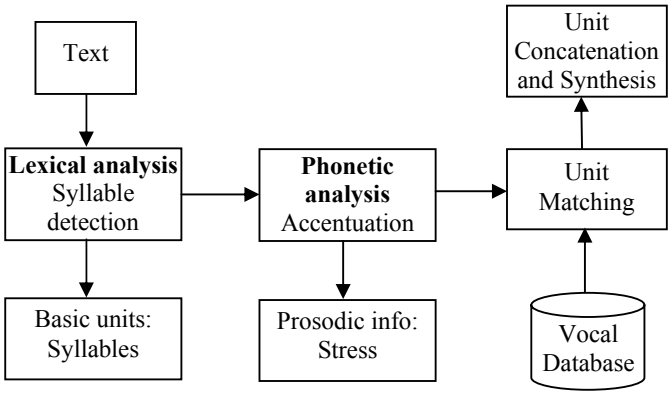

Figure 5. The principle of our syllable-based speech synthesis system.

7. CONCLUSIONS

We have presented in this article a rule-driven approach for construction of a syllable-based TTS system. Special efforts have been done to accomplish the text processing stage. After serious researches in linguistic field, we have designed two sets of rules: one for detecting word syllables and a second set for determining which syllable is accentuated in each word. Even these sets are not complete, they cover yet a good majority of cases. The lexical analyser is entirely based on rules and assures more than 85% correct syllable detection and 75% correct stress detection rate at this moment.

By the advantages of detecting syllables through the rule-driven analyser we can mention: easy accessibility and readability of rules, easy modifying of existing rules and facile insertion of new ones.

Even in this stage of TTS prototype development, the results obtained are encouraging. The synthesized speech is understanding and conserves the quality of recorded units. We have observed that normalization and correct concatenation are essentials for the quality of speech. As future development, we think that F0 adaptive correction in concatenation points will increase the smoothness of the signal. We also consider that completion of syllable database in next stages will bring more accuracy in the syllable generation process.

REFERENCES

1. BURILEANU D. *et al.*, "A Parser-Based Text Preprocessor for Romanian Language TTS Synthesis", *Proceedings of EUROSPEECH'99*, Budapest, Hungary, **Vol. 5**, pp. 2063-2066, September 1999.
2. CONSTANTINESCU-DOBRIDOR G., *Sintaxa limbii române*, Editura Ştiinţifică, Bucureşti, 1994.
3. CIOMPEC, G. *et al*., *Limba română contemporană. Fonetică, fonologie, morfologie*, Editura Didactică şi Pedagogică, Bucureşti, 1985.
4. FREE SOFTWARE FOUNDATION, *Flex - a scanner generator*, http://www.gnu.org /software/flex/manual, October 2005.
5. HUNT A., BLACK A., "Unit selection in a concatenative speech synthesis system using a large speech database", *Proceedings of ICASSP '96*, Atlanta, GA, May 1996, pp. 373-376.
6. LEWIS E., TATHAM M., "Word and Syllable Concatenation in Text-To-Speech Synthesis"**,** *Sixth European Conference on Speech Communications and Technology*, pp. 615-618, ESCA, September 1999.
7. BUZA O., *Vocal interactive systems*, doctoral paper, Electronics and Telecommunications Faculty, Technical University of Cluj-Napoca, 2005.
8. BUZA O., TODEREAN G., "Syllable detection for Romanian text-to-speech synthesis", *Sixth International Conference on Communications COMM'06*, Bucharest, June 2006, pp. 135-138.
9. BUZA O., TODEREAN G., "A Romanian Syllable-Based Text-To-Speech System", *Proceedings of the 6th WSEAS International Conference on Signal Processing, Robotics and Automation (ISPRA '07)*, Corfu Island, Greece, February 16-19, 2007.
10. YOUSIF A., ZURAIDA M., "Text-to-Speech Conversion of Standard Malay", *International Journal of Speech Technology*, **Vol.3**, Number 2, June 2000.