TTS EXPERIMENTS: ROMANIAN PROSODY

Arpad Zsolt BODO[*, **], Gavril TODEREAN[*], Ovidiu BUZA[*]
[*]*Technical University of Cluj-Napoca, Faculty of Electronics, Telecommunications and Information Technology;*
[**]*Sprint Consulting Ltd.,*
email: zsolt.bodo@sprintconsulting.com

Abstract: In a text-to-speech synthesis system, prosody prediction and generation, is one of the most critical tasks. Prosody is composed by the melody, intensity and by the rhythm of speech. The current paper focuses on the Romanian intonation prediction and generation within our TTS system. Experimental results achieved by the development of different prosodic modules are presented in this work. The process is followed from text pre-processing, sentence type determination, through automatic labeling of sentences, till the labeling and transposing of the intonation curves. The creation and storage of relative intonation curves, as the developed different rule systems described in XML are also topic of this work.

Key words: *Intonation; Melody; Labeling; Prosody; Sentence type; Text analysis; Intonation curve; XML rules.*

I. PROSODY IN A TTS SYSTEM

Imposing prosody, in a TTS process, is the result of text analysis and prosody prediction; physically this is done by controlling the F0 frequency, temporal structure and amplitude of speech signal.

The way of prosody imposing is totally dependent of the applied synthesis technique. In the literature of this domain various TTS architectures are detailed. Most of these is modular. The diversity of these architectures has its origins in the variety of the applied synthesis techniques.

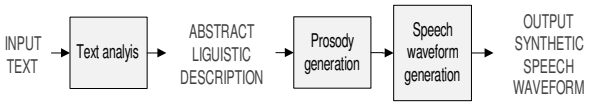

Figure 1. TTS analysis-synthesis process [2], page 94.

While Figure 1 shows a generic TTS analysis synthesis process, Figure 2 presents the architecture implemented by the author.

According to this architecture, a controller is driving the whole process. In this process, prosody generation is preceded by a detailed text analysis. Text analysis, prosody generation and prosody imposing engages several databases that contain the different rules for analysis, or raw data like reference intonation curves, etc. The advantage of this architecture is that in most of the cases, the whole system can be extended and improved without any program code modification. The following chapters will detail the prosody related modules, and their realization.

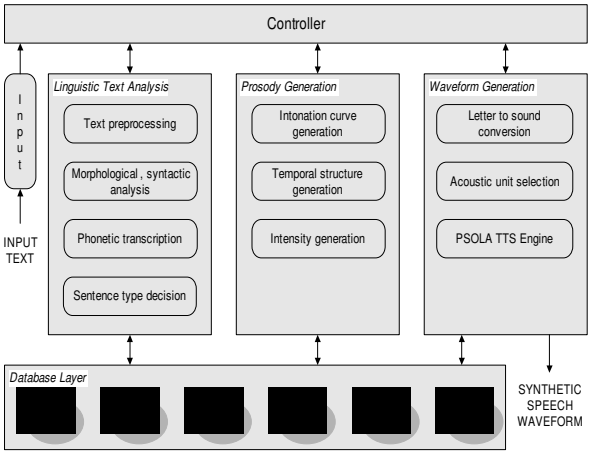

Figure 2 – Implemented TTS Architecture

II. TEXT PROCESSING

Classical TTS systems are realizing prosody generation without learning methods. These prosody modules must therefore rely completely on the input text. That is why the prosody generation preceding text analysis is such a key phase.

In the following we detail the text analysis aspects of prosody generation, and our realizations.

Figure 2 contains the rough structure of a linguistic text analysis module, which is found at the beginning of the analysis-synthesis chain.

Text preprocessing

Manuscript received February 9, 2009; revised April 14, 2009

The written text can be considered as a series of electronically coded characters (ASCII, UNICODE). Its elements are characters of type alphabetic, alphanumeric, separator or other special characters.

Segmentation is the first step of text preprocessing, where the long chain of characters is decomposed to blocks that are easier to handle (e.g. sentences). These blocks are further decomposed to words. The boundary definition of words usually is simple, since these are separated by spaces. The case of sentences is however more complicated, since e.g. the character "." could mean the end of a sentence, but also an abbreviation. Moreover, in some cases could mean decimal point as well.

Normalization is needed because of the following diversity: the set of characters used by an arbitrary text is not limited: may contain numbers (e.g. *1.99; 2; 2.; IV;*), abbreviations (e.g. *Dr.*), special characters (e.g. *%, *, @, etc.*), upper- and lowercases, punctuations or even format characters (e.g. tabulators). Normalization means meaning that the input text is going to be transformed in a series of words, which can be pronounced. The segmented, and then normalized text will contain unambiguously and well defined separated words written lowercase. This way the Romanian sentence:
„ Dl. Prof. Dr. Ing. Gavril Toderean ajunge pe str. Bariţiu la 2 PM."
Will look like:
{[domnul][profesor][doctor][inginer][gavril][toderean][ajunge][pe][strada][bariţiu][la][doi][peem]}
In this structure above, the *{}* mark the sentence boundaries, the *[]* mark the word boundaries like in [2], p. 98. During normalization, each resolved word (unambiguously described) can be assigned with a label showing the type of this resolution (whether this is an abbreviation, number, etc). Information carried by these labels can be useful in later processing.

Most of the TTS systems are using substantial number of rules for handling the different types of text items. Here we present some according to [2]:

The resolution of numbers depends for example on the context:

- The four digit numbers usually mean a year.
- A " ' " followed by two digits might also represent a year, e.g. '06 = 2006
- The phone numbers should be pronounced according to the specific of the language, dialect, but they could be spelled as digits also.
- In case of a sum of money: 25,50 RON = 25 lei şi 50 de bani (Romanian).
- Consequently this text resolution might sometimes need syntactical analysis also. Therefore, some authors handle a part of normalization in a later phase (see Sproat in [7], page 32)

For abbreviations, special characters, one could use a mapping table with items like:

- % = procent; @ = et; Tg.= Târgu; etc .= etcetera (Romanian)
- In case of abbreviations: UTC-N = Universitatea Technică Cluj-Napoca (Romanian)
- In case, when the abbreviation is not present in the mapping table, another solution could be e.g. the pronunciation like a word (the Romanian UTC-N could be "utecene"). If no other solution exists, the spelling phone by phone can still be followed.

However the types of rules mentioned above generally are applicable, in some cases the final resolution should be left to a later phase.

Morphological analysis

Morphemes are the smallest elements of a language having still a meaning. For example the Romanian word "*plimbat*" contains two morphemes: "*plimba*" (the root "to walk") and "*t*" (past). The words are built by one, two or more morphemes. These can be roots, prefixes and suffixes. The morphemes are abstract units, which can appear differently in different contexts. For example see the English "*think*" and "*thought*", or the gender in Romanian of "*acest(a)*" and "*aceast(a)*" For morphological analysis, Giurgiu in [9] page 97 mentions the analysis from left to right and from right to left together to determine the root and the terminations.
It is vital to perform morphological analysis for the phonetic languages (e.g. English).

Phonetic transcription

This is more important in case of phonetic languages, like English, where huge pronunciation dictionaries are used. However, in languages like Romanian, this is also needed. Briefly, the strategy of correct pronunciation of a word is to use the following:
a) Pronunciation dictionary – words, morphemes, rules, etc.
b) Letter-to-sound rules –complex especially for phonetic languages.
c) Lexical accents – a sound depends on its context (e.g. in the Romanian word *"handbal"* the group of sounds [*ndb*] is heard as [*nb*] only).

Lexical accents and word accents

In case of the polysyllabic words only one syllable will carry the **primary accent**; the others will carry no, or less, so called **secondary** accents. In case of Romanian this is usually the before last one. One task of text analysis is to distinguish the so called "functional words" (adverbs, pronouns, etc.) which are showing the relation between the "component words" (content). The identification of the keyword is also important from prosody perspective (Romanian: *nici, sigur, nu, poate, etc.*).

Practical text preprocessing solution

Similar to those mentioned above, a set of rules have been created by the author and have been described in XML (eXtensible Markup Language).

Table 1. – Text processing rules for the example sentence

```
<?xml version="1.0" encoding="UTF-8" ?>
<preprocessingRules>
   <abbreviations>
      ...
      <abbrev text="Dl.">
         <in_words text="domnul"/>
      </abbrev>
      <abbrev text="Dr.">
         <in_words text="doctor"/>
      </abbrev>
      <abbrev text="Prof.">
         <in_words text="profesor"/>
      </abbrev>
      <abbrev text="Ing.">
         <in_words text="inginer"/>
      </abbrev>
      <abbrev text="str.">
         <in_words text="strada"/>
      </abbrev>
      <abbrev text="PM">
         <in_words text="dupa masa"/>
      </abbrev>
      ...
   </abbreviations>
   <numbers>
      ...
      <number value="2">
         <in_words text="doi"/>
      </number>
      ...
   </numbers>
   <symbols>
      ...
   </symbols>
   <alphabet>
      ...
   </alphabet>
   ...
</preprocessingRules>
```

This mentioned set of rules contains:

a) abbreviations,
b) symbols,
c) English alphabet pronunciation,
d) the numbers between 1- 999.

Advantage of description by XML is its simplicity to append new rules. For the normalization of the example sentence mentioned earlier (*"Dl. Prof. Dr. Ing. Gavril Toderean ajunge pe str. Bariţiu la 2 PM"*) the subset of the applied rules are shown in Table 1.

III. SENTENCE TYPE DECISION. SYNTACTIC ANALYSIS

Jinga [1] provides a detailed description of the Romanian intonation system by sentence intonation patterns types for declaratives (e.g. descending neutral, of finality and continuity, descending emphatic, predicative, etc.) and interrogatives (e.g. ascendant, emphatic, with "vocative", ascending rhetoric, ascending elliptical, echo questions, descendent, partial questions with more interrogative words, negations, short elliptical partial questions, ascending-descending interrogative, descending-ascending interrogative, tag questions, complex questions, etc.).

In order to perform a detailed study of those presented in [1], each presented pattern type had been reconstructed by the author (see Figure 3, 4, 5). As example the *neutral declarative descending intonation pattern* is presented. According to [1] the intonation pattern of a declarative short phrase, at first look, has an ascending-descending aspect:

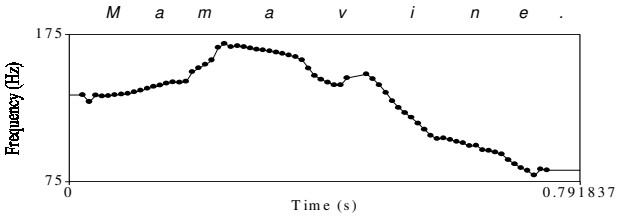

Figure 3. Intonation of Romanian sentence "Mama vine" – [1], p. 15

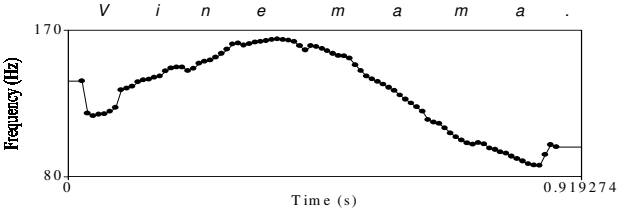

Figure 4. Intonation of Romanian sentence "Vine mama" – [1], p. 15

But in reality, the two movements of the tone are asymmetric: the end reaches a lower value then the starting one. In case of lengthening of such a prototype sentence, what is actually lengthened is the second part (descendent) of the pattern:

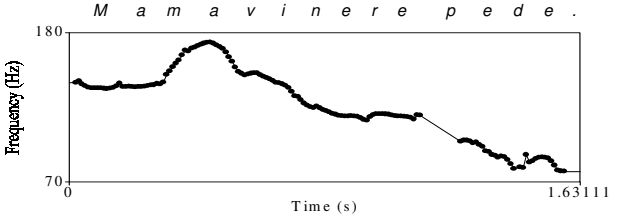

Figure 5. Intonation of Romanian sentence "Mama vine repede" – [1], p. 15

According to [1], this descendent pattern is universal in the majority of languages.

After performing the text analysis described by the previous chapter, sentence type identification follows. While Jinga describes the intonation pattern of different sentences, opens a great opportunity for us to build a system of Romanian sentence types. It is quite easy to find out about a sentence whether it is declarative, interrogative or imperative. But for prosody determination more information is needed. During our studies, first of all we have identified the sentence types based on [1]. For each type we have defined an ID. For example *S1xx* is the set of declarative sentences, *S2xx* is the set of interrogative sentences, *S3xx* is the group of imperatives, etc. Table 2 presents some of the identified sentence intonation pattern types.

Table 2 – A few of the identified Romanian sentence intonation patterns

ID	Intonation Patterns
S111	Declarative, descending neutral
S121	Declarative, continuative and finality
S122	Declarative, enumerative
S123	Declarative, with continuous completion
S124	Declarative, ascending, positively raising
S132	Declarative, ascending negative
…	…
S211	Interrogative, yes-no questions
S212	Interrogative, starting with negative word
S219	Interrogative,
S213	Increasing pattern, starting with interrogative word
S215	Interrogative, elliptic
S217	Interrogative with separate interrogative word
S221	Interrogative word at the beginning
S225	Interrogative word within the sentence
S224	Interrogative repeating sentence
S231	Interrogative ascending descending
S241	Interrogative descending ascending
…	…
S320	Imperative, indirect
S330	Imperative, with keyword
…	…
S400	Declarative, with interruptions

For the majority of these sentence types it was possible to construct some identification rules. This algorithm analysis the written text from several aspects in different levels:

1. On the first level we analyze the end of a sentence (*“.”, “?”, “!”, “…”, etc*); this the set of possible types is substantially reduced.
2. On the second level, the analysis is performed by punctuation and bywords. The punctuations (like: *“,”, “;”, “:”, “-”, etc.*) are great influencing factors of the sentence monotony of the sentence and are causing a change of intonation. Therefore if these exist, their place in the sentence is determined. At “,” (coma) the sentences usually reach their highest tone, while at “-” and “…” the lowest.
3. On the third level of the analysis the words are examined, by their kind, type, place in the sentence, etc. In Romanian, words like *“nu”, “nimeni”, “nimic”, “nici”* are used by negative sentences, this way their presence is helping at the sentence type determination. The presence of adverbs, pronouns, interrogative words is carrying essential information. Unfortunately it is not always possible to find such keywords.
4. The fourth level is not reached by each sentence, except those, which could not be identified by the rules above. In this case phrase level analysis is performed. For example an interrogative sentence is a so called Yes-No question, if this sentence is followed by answers like *“Da.”* or *“Nu.”*.

The diagram from Figure 6 illustrates the described algorithm by determining the type of a particular sentence. Step by step the intermediate results, and at the end the final decision is shown.

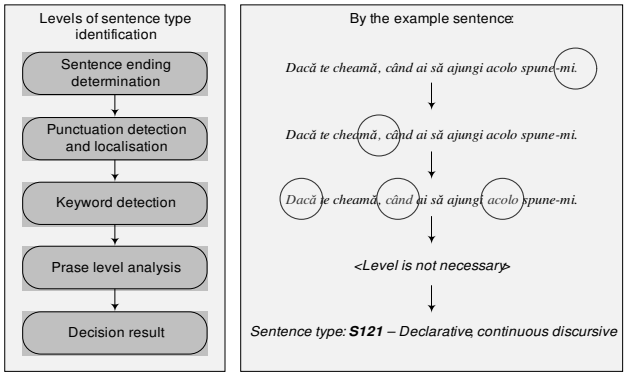

Figure 6 – Steps of sentence type determination algorithm via an example

The rule system serving this algorithm has similar structure described in XML. Its tree structure has the same levels as the algorithm, and its leaves are the result of sentence type decision. For a better understanding, see Figure 7.

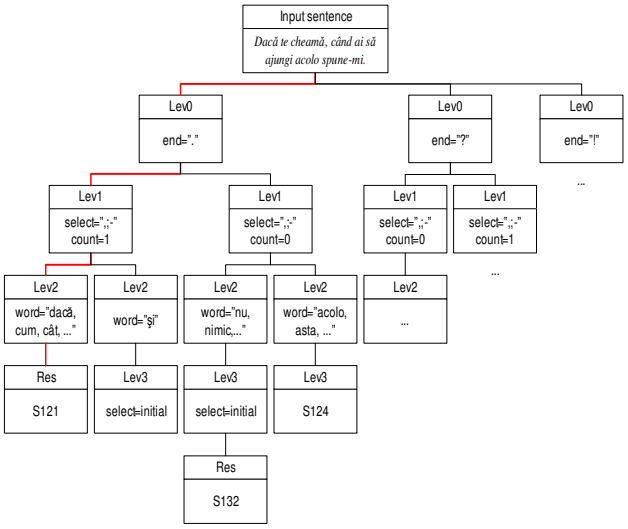

Figure 7 – Part of XML tree for sentence type determination algorithm

The *SentenceRules.xml* contains the rules for sentence type determination. Each line of this file is a node, containing the level and the attribute. The line *“<lev0 end=”.”>”* describes the level *lev0* and the attribute *end* with attribute value *”.”*. The level naming convention is ***lev0, lev1, lev2,* etc**., except the level, of the result, named ***res***. After identifying the level, like *lev0*, we check the argument. If the argument is the keyword *end*, obviously this is about the ending of the sentence. Based on the attribute value, like *“.”*, is possible to decide the sentence type as declarative. Afterwards, the algorithm goes to the next level, *lev1*, and so on. In case on a certain level no result is found, a step back is taken.

As argument several keywords have been used: ***select*** (selects the value and saves it), ***count*** (the number of

occurrence of the selected value), ***position*** (position of the selection), ***word1_n, val, distbegin, distend, len***, etc.

IV. AUTOMATIC TEXT LABELING

After sentence type detection, the sentences need to be labeled on those words, locations, where the intonation curve is to be influenced. A structure will store these words, within words the proper syllable and the label name. It is very important to identify each existing label, because these will be mapped to the labels of the corresponding intonation pattern – see following chapter.

The simplest labels are the ***min*** and ***max***, corresponding to the minimum and maximum values of the sentence intonation curve. In a sentence might exist more minimum and/or maximum values; in this case each of them will be labeled. In case the 70% of the utterances, there are not more, than two minimum and two maximum values in their intonation curve. In case of an enumeration, where each enumerated item's last syllable is emphasized, a new label will be introduced, ***es*** (enum start) and ***ee*** (enum end), to mark the beginning and the end of it.

A dedicated **automatic labeling module** is able to tag each identified, known sentence type with these labels. This module will save in a list all the high and low toned words and their label name. For the labeling of sentences a three level XML tree had been built: *IntonRules.xml*. The concept is similar to that from the sentence type determination. Figure 8 presents a part of the XML tree for the sample sentence labeling.

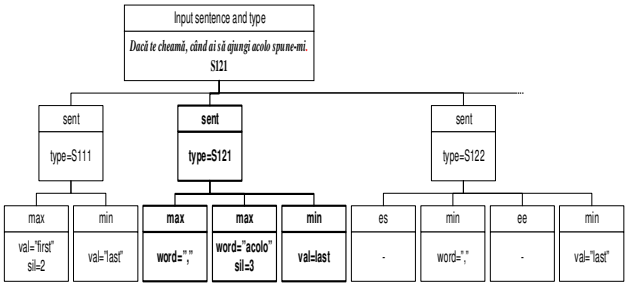

Figure 8 - Part of XML tree for automatic sentence labeling algorithm

The tree structure is simple. After the enumeration of the sentence, on the next level the corresponding labels and keywords are listed. Its advantage is the simplicity of its expansion. For the description of labels some keywords have been introduced: ***type*** (sentence type ID), ***word*** (the place of the label), ***val*** (first-last for the punctuation, etc.), ***sil*** (the number of syllable of the word).

The result is such a list, which contains the marked words, their syllable and label.

V. REFERENCE INTONATION CURVE CREATION AND LABELING

For each identified sentence type there are corresponding intonation patterns. These patterns are so called *reference patterns*, which can be easily built and stored.

In order to generate these known patterns typical speech utterances have been recorded followed by the computation of their intonation curve. These intonation curved have been saved in **.IntonCurve* files (e.g. *S111.IntonCurve* storing the reference intonation pattern of the sentence type S111).

Further processing these stored reference curves, they are labeled at those points, where the certain modifications will occur during the synthesis phase. These points are the minimum, maximum values, beginning, end of an enumeration, etc. The same label set is as in case of sentence labeling in the previous chapter. This tagging is done offline, manually and the result is stored in **.tag* files (e.g. for reference intonation pattern S111 the tag file will be *S111.tag*). The tag files contain pairs of sample number (0-199, because the reference patterns are stored with 200 samples) and label name.

One of the later steps in the process of TTS synthesis is to impose the proper generated intonation curve. The reference intonation curve proportions, as far as the label distances concerned, are intrinsic result of the original sentence from its generation and recording phase. Because this system is a non-restricted one, the sentence to be synthesized could be of any length, with its labels in arbitrary locations. With other words, however we are talking about the same sentence types, the reference intonation curve and the desired one will never match, and this way cannot be imposed. To adapt the reference pattern to that particular sentence, which is to be synthesized the reference pattern is broken to sections limited by the curve labels. These sections are transposed to the sections limited by the sentence labels, and their length is stretched or compressed. The actual lengths at this moment are calculated from the sum of concatenated diphones lengths. The frequency values are also normalized and adjusted. Figure 9 presents the reference intonation curve and its adjustment to the synthesized sentence.

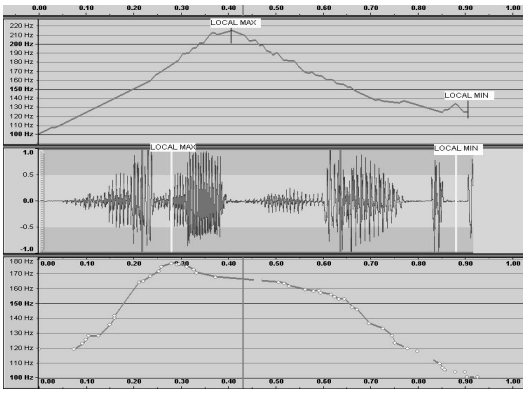

Figure 9 – Reference intonation pattern adjusted to synthesized sentence

The generated and transposed intonation curve, together with the phonetic transcription, will serve as input for the next module, the synthesis engine. This engine, using the PSOLA (Pitch Synchronous Overlap Add) technique will generate the waveform according to these inputs. However this topic is out of scope of current paper.

VI. CONCLUSIONS

To try to define rules based on Romanian intonation pattern types described by linguists, turned to be a good approach. The idea of having reference intonation patterns and in the same time following a precise transposing resulted in a quite natural speech. To focus within prosodic constituents more on intonation had a high benefit. Of course this does not mean that intensity and temporal patterns would not be important, but our experiences confirm that the majority of prosodic information is carried by intonation.

The investment into the realization of a development environment turned to be a good idea. From technical perspective, a good architecture and a good development environment makes the research work to run smoother. Since we have an easy extendable system, the new ideas can be tested and added easier.

The flexibility of the TTS engine, the quality of its result proves that PSOLA was a good choice.

As next steps, we plan to increase each of our databases, to add new rules to our text processing XML files, new intonation patterns, new rules to tag descriptors, fine-tuning of our diphone database. Having now the phrase level prosody, micro prosody should be focused more (like word accents).

REFERENCES

[1] L. Dascălu-Jinga, *Melodia vorbirii în limba română,* Academia Română, Univers Enciclopedic, Bucureşti, pp. 27-31, 2001
[2] J. Holmes, W. Holmes, *Speech Synthesis and Recognition*, 2nd Edition, Taylor & Francis, London, pp.93-108, 2001
[3] W.D.E. Verhelst, *An implementation of the PSOLA/KDG Waveform Synthesis Technique*, Eindhoven, Netherlands, pp. 1-9,1990
[4] D. Hirst, A Di Cristo, *Intonation Systems, A Survey of Twenty Languages*, Cambridge University Press, pp. 1-3, 1998
[5] Huang, Acero, Hong, *Spoken Language Processing*, Prentice Hall PTR, pp 689-793. 2001
[6] E. Keller, *Fundamentals Of Speech Synthesis And Speech Recognition*, Switzerland, pp. 23-41, 1994
[7] R. Sproat, *Multilingual Text-To-Speech Synthesis*, Kluwer Academic Publishers, pp. 141-150, 1998
[8] G. Olaszy, G. Nemeth, P. Olaszi, etc., "Profivox – A Hungarian Text-to-Speech System for Telecommunications Applications", *International Journal of Speech Technology* (IJST) – Vol. 3, No. 3/4, Kluver Academic Publishers, pp. 201-215, 2000.
[9] M. Giurgiu, L. Peev, *Sinteza din text a semnalului vocal,* Vol. I. *Modelare acustică şi fonologică*, Cluj-Napoca, Risoprint, pp. 142-167, 2006
[10] A. Ferencz, *Contribuţii la dezvoltarea sintezei „text-vorbire" pentru limba română*, Teză de doctorat, Cluj-Napoca, pp. 52-56, 1997
[11] T.-CS. Kovacs, *Disertaţie de masterat*, Cluj-Napoca, 1997
[12] A Zs Bodo, *Experiments for prosody modification using the Nonlinear Springing Method*, Trends in Speech Technology, SpeD05, Cluj-Napoca, , pp. 177-181, 2005
[12] A Zs Bodo, O. Buza, G. Toderean, "Acoustic Database for Romanian TTS Synthesis. Design and Realisation Results (I)", *Acta Technica Napocensis – Electronics and Telecommunications*, ATN Vol. 48, Nr. 2/2007, pp. 24-31, 2007
[13] A Zs Bodo, O. Buza, G. Toderean, "Realisation Results of a speech Synthesis Development Environment", *Acta Technica Napocensis – Electronics and Telecommunications*, ATN Vol. 48, Nr. 2/2007, pp. 32-37, 2007
[14] I. Kounty, G. Olaszy, P. Olaszi, etc., "Prosody prediction from Text in Hungarian and its Realisation in TS Conversion", *International Journal of Speech Technology* (IJST) – Vol. 3, No. 3/4, Kluver Academic Publishers, pp. 187-200, 2000.
[15] D. O´Shaughnessy, *Speech Communications, Human And Machine*, IEEE Press, pp. 337-366, 2000