

# Arabic Phonetic Web Sites Platform Using VoiceXML

**Bassem KOSAYBA**

Departement of Software Engineering & Information Systems  
Damascus University, Syria  
script.java@gmail.com

**Hasan Alkhedr, Firas Jdeed**

5<sup>th</sup> year Students – Albaath University, Homs, Syria

**Amer Shriedi, Eman Al-mozaien**

5<sup>th</sup> year Students – Mamoun University of Science and Technology (MUST)

**Abstract-** In this paper we discuss a new important technology which is a VoiceXML the markup language for browsing sites or providing services among human speech over the network (internet or other). We suggest a platform to add Arabic support to this technology. We introduce this technology and its methodology (how to work). Since that this technology requires the user to own some major components such as VoiceXML browser, Arabic TTS and Arabic ASR, we describe these components in some details. The paper firstly introduces the VoiceXML, then it explains the architecture of the VoiceXML browser in details, after that it discusses the specifications of Arabic language, next it discusses the method of building Arabic TTS and Arabic ASR and their architecture.

**Keywords-** VoiceXML, Arabic language, Arabic TTS (Text-To-Speech), ASR (Automatic Speech Recognition).

## I. INTRODUCTION

VoiceXML is a markup language derived from XML "eXtensible Markup Language" for voice applications. The power of VoiceXML is that it standardizes voice application development by leveraging the full set of available web development tools and techniques. For example, developers can write a single server-side program that can be used to display stock quotes on the web or read-back stock quotes over the phone.

This program would use the same business logic code for getting the quotes, but would have code for two user interfaces one for speech and one for HTML. Major goal of VoiceXML is to bring the advantages of web-based development and content delivery to interactive voice response applications.

The existing web infrastructure was designed for traditional desktop browsers and not for hand-held devices. The data in the web is stored in HTML format which cannot be delivered to mobile devices. The only acceptable way to present data to

devices like mobile phones or traditional phones, is audio. Certain voice browsers are capable enough to process VoiceXML content and produce the output in the form of audio using components such as speech recognizer (ASR) and speech synthesizer (TTS).

VoiceXML is designed to create audio dialogs that feature digitized synthesized speech, recognition of spoken as well as telephone keypad input, and the recording of spoken input. VoiceXML allows developers to build dynamic web sites including voice (speech) services which allows us to interact with web applications via voice and speech.

VoiceXML facilitates the browsing ability for crippled people and even for normal ones, as they can browse the Internet while they're driving, shopping...etc. Unfortunately the Arabic language is not supported yet in the VoiceXML technology, even so its one of the most spoken language over the universe.

## II. OBJECTIVES AND APPROACH

To develop VoiceXML editorial and browsing tools with Arabic language support. Currently the browsing and editing tools are both designed by XML-based markup languages without the function of traversing voice to text and vice versa in Arabic.

Our work is to build this platform including our developed Arabic TTS (Text-To-Speech), and Arabic ASR (Automatic Speech Recognition) and merging them with voice xml.

Our platform will accept Arabic voice requests from clients, and responds to their requests according to the web server. The platform will also provide Arabic phonetic speech recognition and Text-To-Speech as its not included in the standard VoiceXML specification.

Our platform should provide support for voice service browsing from traditional devices such as computers and for hand-held and mobile devices, so we have to take care of specifications of these devices. For this reason, there are two

ways to develop the platform. First we can make the platform as distributed as possible as in figure 1 wherefrom we can place base components of the platform on the client-side (devices) as ASR and TTS so we will convert each audio request file to VoiceXML file using speech recognizer and transfer the resultant file over the network to the server which issue the request and responds it as VoiceXML file send to the client, and this respond file will be converted to voice (audio) by speech synthesizer located on the client side.

Second, we can make or develop a centralized platform including all components on the server-side. In this case, ASR and TTS will be located on the centralized telephony server (PSTN) and the request and response will be transferred over the network as audio streams and analog signal. Here we can develop more and more voice applications to support mobile devices and telephony services.

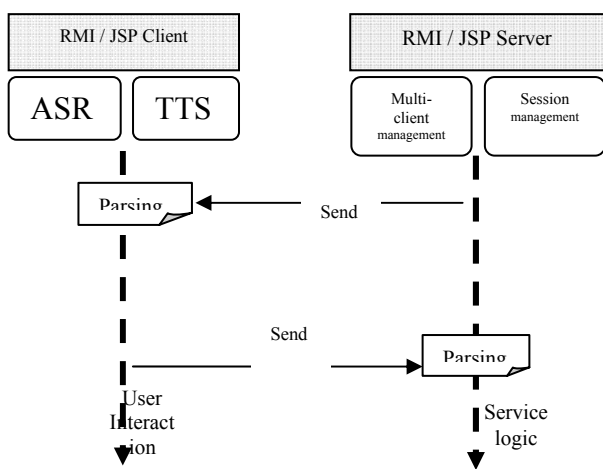


Figure 1. distributed platform overview

In our work we chose the first architecture which allows us to benefit from the existing components so that we don't need to rebuild these components *"the goal is not to invent the wheel again, but to develop it"*. By using this method we assure that we always work according to a predefined and trustworthy standards.

### III. FRAMEWORK

#### A. System overview:

We can consider that the VoiceXML platform is a middle layer between the web server and the human. This means that the platform acts as internet browser but it replaces viewed pages by voice and user input by user speech i.e. the interaction between end user and our platform is made by speech.

Figure 2 shows a high level overview of the system who has the following components:

- A document server (i.e. a web server): processes requests that received from a client application (called the VoiceXML interpreter context). The server produces VoiceXML documents in reply. These documents are read by a VoiceXML interpreter, which is inside a VoiceXML interpreter context.
- The VoiceXML interpreter context: monitor user inputs in parallel with the VoiceXML interpreter. the VoiceXML interpreter context may be responsible for detecting an incoming call, acquiring the initial VoiceXML document, and answering the call, while the VoiceXML interpreter conducts the dialog after answer.
- The implementation platform is controlled by the VoiceXML interpreter context and by the VoiceXML interpreter. The implementation platform generates events in response to user actions (e.g., spoken or character input received, disconnect) and system events (e.g., timer expiration). Some of these events are acted upon by the VoiceXML interpreter itself, as specified by the VoiceXML document, while others are acted upon by the VoiceXML interpreter context. Implementation platform includes converting voice input from the microphone on client PC or from any other client device into Arabic text by ASR component (Automatic Speech Recognizer) , and converting the text that is coming from the VoiceXML document to voice and specifically to Arabic speech that the client can hear and understand by the TTS component (Text To Speech).

The whole scenario of the interaction is as following: the server sends the VoiceXML document to the interpreter which takes the document and sends the text between <prompt> tags or other tags that require the browser to say something to the TTS and the client listens the text between

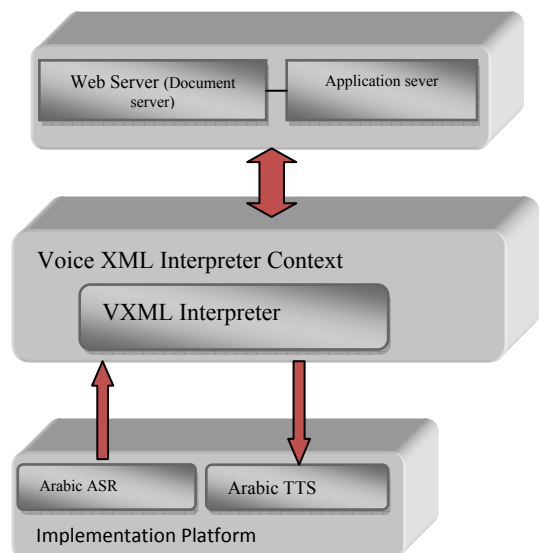


Figure 2. system components

these tags, then the client speaks what he wants, and his/her speech will be passed to the ASR which convert it to Arabic text and send this text to the interpreter platform which then make a new VoiceXML document with the URL of the next document and send it back to the server, after that the server will send the appropriate document to the interpreter.

From the previous section, now we can say that the system consists of the following components as shown in (figure 2):

- VoiceXML interpreter: the role of this component is to manipulate vxml files from the server and work with speech by connecting to the TTS (Text To Speech) and ASR (Automatic Speech Recognition).
- Arabic TTS: which will convert text received in vxml file to Arabic speech.
- Arabic ASR: which will convert user Arabic speech to text.
- Application server: or web server which store the vxml documents each document with unique identifier (URL).

**B. Detailed architecture:**

As mentioned above, the system consists of number of components and each of these components can be implemented in more than one way. We'll discuss detailed specifications of some of these components in the following sections:

*1) VoiceXML interpreter:*

This is the base component in the system and most of our work is in this component. There are several free (open source) VoiceXML interpreters but there is no interpreter provide interaction with the user in Arabic language and this is what we exactly need, so we will work for adding Arabic support to an existing interpreter by adding two APIs to the interpreter, one for managing Arabic ASR and the other for managing Arabic TTS. Or we can build a full new Arabic VoiceXML platform that can provide these functions.

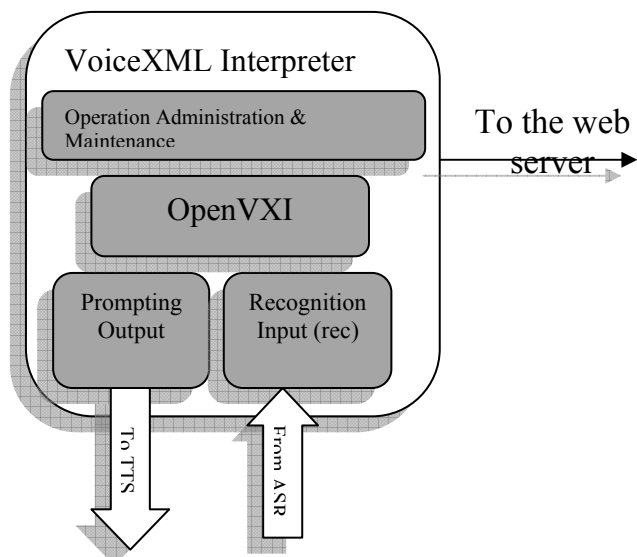


Figure 3. Interpreter architecture

The core of the VoiceXML Browser is the OpenVXI open source VoiceXML interpreter and its associated internet

components for XML parsing The browser consists of the following components (figure 3):

*a) An operations administration and maintenance (OA&M) system and main process:*

This collection of tools is responsible for system management, session management and error reporting. This critical component also invokes the speech browser within a thread it creates to begin execution.

*b) The OpenVXI:*

This is the component that interprets the VoiceXML markup and calls into the implementation platform to render the markup. The VXI is tied directly into the Xerces parser. The OpenVXI uses the SAX version of the Xerces parser and does VoiceXML validation for every document. The VXI uses the SAX parser to convert the VoiceXML to an internal binary format that is then available for caching. This leads to a significant performance improvement [3].

*c) Prompting output:*

VoiceXML 2.0 prompting is considerably more complex than playing a set of audio files and TTS prompts. The prompting implementation should be able to:

- Support fetch audio if no other prompt is playing.
- Synthesis more than one speech signal from input text and choose best (n-Version).
- Support SSML including interleaving TTS and audio for playback.
- Handle fetch failures and swapping to TTS when audio fetches fail.

When the interpreter encounters a prompt component that contains SSML, it passes the text to the TTS. The Queue method of the interface provides this delegation. The Queue method takes the text. Queue then blocks until the data is fetched, or the stream is started.

*d) Recognition Input (rec component):*

The rec component is responsible for processing user input. An implementation of the rec interface should be able to:

- Support Arabic speech recognition against multiple parallel grammars.
- Allow for both speech and DTMF entry.
- Return one or more recognition with corresponding confidence scores (n-Version).
- Implement the 'builtin' grammars for Arabic language for simple types (e.g. date, time, and currency).
- Return the waveforms from recognition utterances and recordings.

Grammars may be specified within VoiceXML directly within the <grammar> element or indirectly. In the second case, the text serves a dual purpose of generating text-to-speech enumerations and speech grammars. The corresponding grammar must be generated within the rec component. The implementation of the rec component must fetch the desired grammar URI and any dependent URIs that are included via the grammar import directive.

The NLSML (Natural Language Semantic Markup Language) standard from W3C is the standard using to build

complex grammars where we can assign or catch more than one data value from each utterance and one complex grammar.

2) *Arabic TTS & Arabic ASR*

Building TTS system or ASR system is very language dependent. It requires a deep analysis of the specification and characteristics of the target language so we firstly introduce the Arabic language in some detail, and then we describe the TTS and ASR.

a) *The Arabic Language:*

The Arabic language belongs to the Semitic group of languages which also includes Hebrew, Farsi and Amharic. Arabic is ranked as number four among the world’s major languages according to the number of native speakers.

The Arabic language contains 29 letters (hamza is the 29th letter). There are 6 vowels in Arabic, 3 short and 3 long and there are 2 semi-vowels, which are diphthongs. Arabic short vowels are written with diacritics placed above or below the consonant that precedes them these short vowels are fatha, damma, and kasrah. And Arabic long vowels are (alef ا , waw و ,and yaa ي).

Appendix A shows the Arabic alphabet and how its letters are classified.

The Arabic alphabet is written from right to left and there is no difference between upper and lower case. Most of the letters are attached to one another and they vary in writing whether they connect to preceding or following letters. Arabic short vowels are not written, Therefore the reader must have some knowledge of the language. Short vowels are marked only where ambiguity appears and cannot be resolved simply from the context. The written Arabic is a language of syllable length, rather than accent or stress furthermore, all syllables should be given their fully length without slurring any letter. This means that one should not emphasis any syllable at the expense of another. In the Arabic language there are two kinds of syllable, short and long ones.

All syllables have a single onset C followed by a long or a short vowel. Short vowels are denoted by "V" and long vowels are denoted by "V:". The short syllable, CV, consists of a consonant with a short vowel. For example the word "كُتِبَ" (he wrote) consists of three short syllables. These syllables should be pronounced in an even and equal way. The long syllables contain a vowelled consonant (consonant with short vowel) followed by an unvowelled letter (consonant or long vowel). So long syllables denoted by: CVV consonant with short vowel followed by long vowel, or CVC consonant with short vowel followed by consonant.

b) *Arabic Text-to-Speech System (TTS):*

Text-to-Speech is the process of converting a written text into artificial speech. It is computer-based program in which the system processes the text and says it aloud. The system consists of two major modules:

- The Natural Language Processing Module (NLP) is able to produce files with a phonetic transcription of the text, together with the desired intonation and rhythm.
- The Digital Signal Processing Module (DSP) transforms the symbolic information set receives from the NLP module into speech.

Each one of the previous module include number of sequential operations each of them depends on the previous one as shown in figure 4. The following sections describe these operations.

1. *The Natural Language Processing module (NLP):*

The NLP module consists of three processing stages: Text Analysis, Automatic Phonetization and Prosody Generation. The first stage, the text analysis, consists of four categories:

1.1 *Text Analysis Module:*

This stage consists of four main operations:

- A pre-processing module, where the input text is organized into lists of words. the first step in the text analysis is to make chunks out of the input text –i.e. tokenizing it. There are many tokens in a text, that appear in a way where their pronunciation has no obvious relationship with their

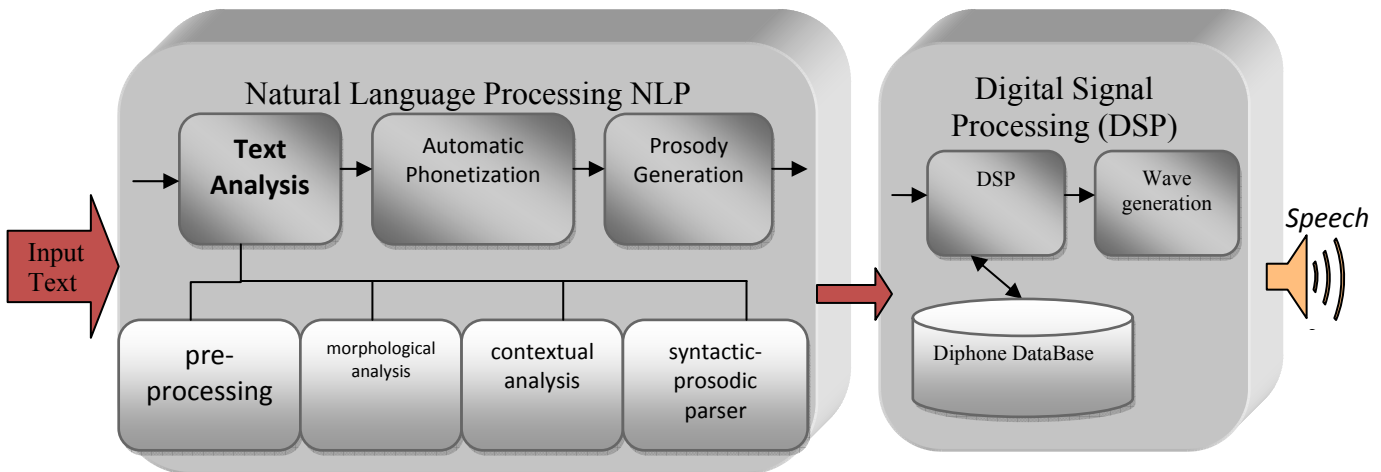


Figure 4. Text-To-Speech System Architecture

appearance [6]. Such as abbreviations (as ص in Arabic, which may be said as "صباحاً" in the morning), and numbers. Apart from tokenization, Normalization is needed where a transformation of these tokens into full text is done.

- A morphological analysis module: this module retrieves the morph (root) of some words in the text.
- A contextual analysis module that considers word in their contexts. This is important to be able to reduce possible part-of-speech categories of the word by simple regular grammars by using lexicons of stems and affixes [6].
- A syntactic-prosodic parser, where the remaining search space is examined and the text structure is found. The parser organizes the text into clause and phrase like constituents. After that the parser tries to relate these into their expected prosodic realization[6].

### 1.2 Automatic Phonetization:

Where the words are phonetically transcribed. In this stage, the module also maps sequences of grapheme (some characters) into sequences of phonemes with possible diacritic information, such as stress and other prosodic features, that are important to fluency in speech and natural sounding speech.

### 1.3 Prosody Generation:

Where certain properties of the speech signal such as pitch, loudness and syllable length are processed. Prosodic features create segmentation of the speech chain into groups of syllables. This gives rise to the grouping of syllables and words in larger chunks.

## 2. The Digital Signal Processing module (DSP):

In this module a transformation of the received symbolic information from the NLP module into speech is done. Concatenative Synthesis is the most technique that used today, where segments of speech are tied together to form a complete speech chain. The speech output is produced by coupling segments from the database to create the sequence of segments. This technique requires a bit of manual preparation of the speech segments. There are two categories within this method, diphone and unit-selection synthesis. The main difference between the two types of Concatenative Synthesis lies in the size of the units being concatenated. Both methods store the pre-recorded speech units in a database, from which the concatenation originates. Those parts of utterances that have not been found, processed and stored in the database are built up from smaller units.

A diphone synthesis uses a minimal speech database containing all the diphones occurring in a given language. Diphones are speech units that begin in the middle of the stable state of a phone and end in the middle of the following one. The number of diphones depends on the possible combinations of phonemes in a language (for Arabic, there are 38 phoneme, so the maximum number of diphones is  $38*38=1444$ ) The basic idea is to define classes of diphones, for example: vowel-consonant, consonant- vowel, vowel-vowel, and consonant-consonant. The syllabic structure of Arabic language is exploited here to simplify the required diphones database. In diphone synthesis, only one example

of each diphone is contained in the speech database. In order to build a diphone database, the following questions have to be answered and determined: What diphone pairs exist in a language and what carrier words should be used? The answer for these questions are very language independent.

After deep search in the TTS systems that supports Arabic or not, we choose to use a MBROLA TTS system in our platform. MBROLA project is a TTS system that has two Arabic voices (ar1, ar2). The MBROLA project was initiated by the TCTS Laboratory in the Faculté Polytechnique de Mons, Belgium see [8]. The main goal of the project is to have a speech synthesis for as many languages as possible. The MBROLA speech synthesizer is based on diphone concatenation the most used technique of synthesis today. The advantages of using MBROLA TTS are the next:

- It provides two Arabic voices and two Arabic databases (ar1, ar2). And we can build our own voice and database that conform to MBROLA tools.
- We can use MBROLA Arabic databases as a voice for FreeTTS system which is a TTS system written entirely in JAVA.
- MBROLA databases for Arabic provide support and voices for a full phone set of Arabic language including characters such as (ح, خ, ض, ص).

But MBROLA has an disadvantage that it forces us to write any Arabic text not using Arabic alphabet but by using of other alphabet (English). But in real, we can solve this challenge by building a simple module converts each Arabic letter from the input text into its corresponding letter according to that used in MBROLA.

We can use another TTS systems, such as festival, but we prefer MBROLA with FREETS because it is platform independent, whereas festival works on the UNIX platform.

### c) Arabic Automatic Speech Recognition System (ASR):

Automatic Speech Recognition is a process of converting the speech on the microphone into text format. ASR process contains the following modules as shown in figure 5:

#### 1. Feature Extraction:

Speech acquisition begins with a person speaking into a microphone. This act of speaking produces a sound pressure wave that forms an acoustic signal. The microphone or telephone receives the acoustic signal and converts it to an analog signal that can be understood by an electronic device. Finally, in order to store the analog signal on a computer, it must be converted to a digital signal. And it includes following steps:

- Pre-emphasis: which include reduce the range of suffers and noise by implying FIR filter on the signal[14]:  

$$H(z) = 1 - az^{-1} \quad 0.9 \leq a \leq 1.0$$
- Endpoints detection: which include detecting start and end of each word by determining the front and end thresholds of the signal.
- Frame blocking: dividing into overlapping frames of 20ms every 10ms. The speech signal is assumed to be stationary over each frame.

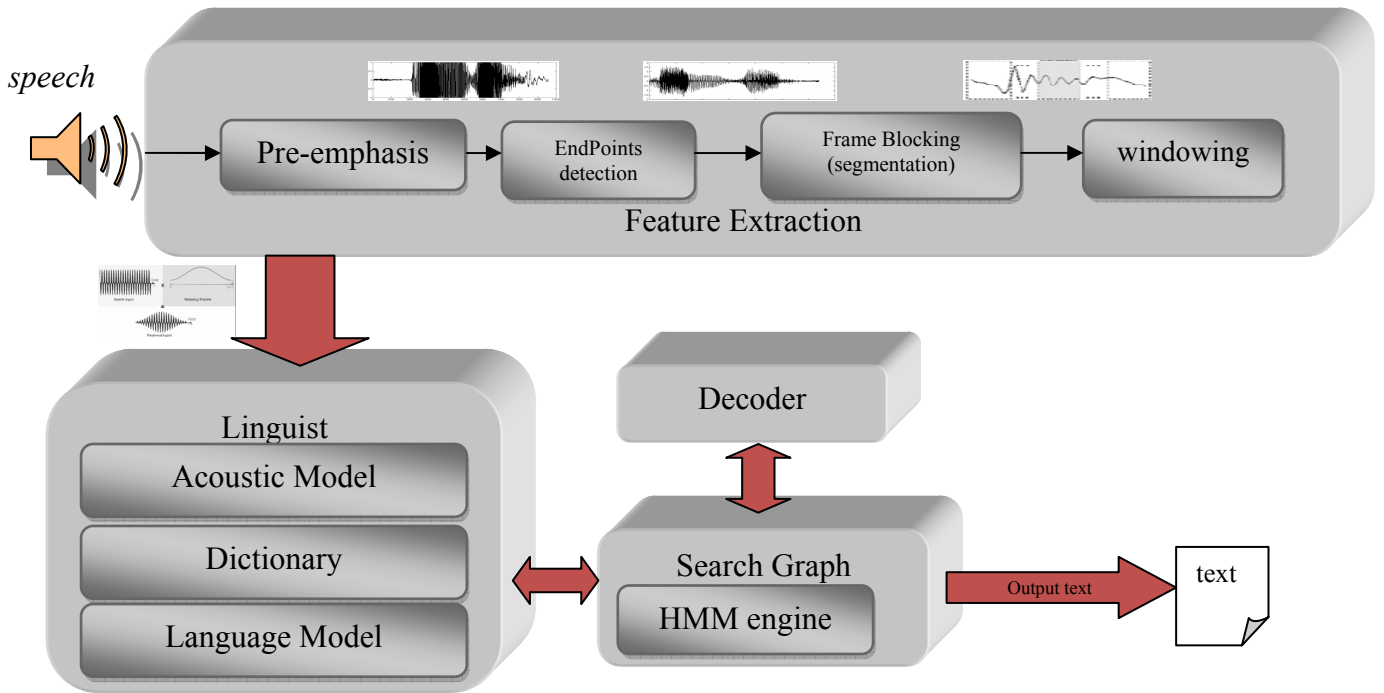


Figure 5. Automatic Speech Recognition System Architecture

- **Windowing:** To minimize the discontinuity of a signal at the beginning and end of each frame, we window each frame to increase the correlation of the linear predictive coding (LPC) spectral estimates between consecutive frames. The windowing tapers the signal to zero at the beginning and end of each frame [14]. A typical LPC window is the Hamming window of the form:

$$w(n) = 0.54 - 0.46 \cos \frac{2\pi n}{N-1} ; \quad 0 \leq n \leq N-1$$

## 2. Linguist:

Or knowledge base, it provides the information the decoder needs to do its job. It is made up of three modules which are [11]:

- **Acoustic Model:** Contains a representation (often statistical) of a sound, created by training using many acoustic data.
- **Dictionary:** It is responsible for determining how a word is pronounced.
- **Language Model:** It contains a representation (often statistical) of the probability of occurrence of words.

## 3. Search Graph:

The graph structure is produced by the linguist according to certain criteria (e.g., the grammar), using knowledge from the dictionary, the acoustic model, and the language model [11]. The system uses HMM (Hidden Markov Model) which makes the search faster in the dictionary according to the HMM states.

## 4. Decoder:

It is the main bloc of the system, which performs the bulk of the work. It reads features from the front end, couples this with data from the knowledge base and feedback from the application, and performs a search to determine the most

likely sequences of words that could be represented by a series of features [11].

After this explanation of the ASR systems, we choose SPHINX which is a speech recognizer written entirely in the JAVA programming language. Now, we don't have a full Arabic recognition system building on SPHINX, but we attempt to train SPHINX to recognize Arabic words. We have this problem, because we don't have an acoustic model for an Arabic language. Also we don't have a dictionary with large vocabularies for Arabic words and each pronunciations.

For this reason, in this work we attempt to recognize Arabic speech depending on the English models and pronunciations.

We have another option to train the system to recognize a speech by using MATLAB that provides us with functions can train a neural network, but by this way each word needs a specific network (Elman Network), so the number of words will be limited.

## IV. CONCLUSION

We have started our works a few weeks ago, and we expect that we will achieve a full work in the next few months. In the end of the work we should be able to produce a platform that provides Arabic language support to VoiceXML browsers which will help more than 200 million Arabic people to use this new technology that will spread largely in the few next years. Not only our framework will provide this, but also it motivates researchers, developers and programmers to build full and high quality Arabic TTS and ASR including full Arabic diphone database, Arabic language model, dictionary, and acoustic model.

## V. FURTHER WORK

In the current work we don't provide a sufficient support for telephony services and mobility applications, so we plan to achieve a specific Arabic platform for such that devices.

Now, one of our main goals is to build full Arabic diphone database supports all specifications of Arabic characters, pronunciations, morphology, and prosody. This database should aim developers to build Arabic dictionary that show all characters in the Arabic language with its exact suitable pronunciation and to build language and acoustic models for Arabic.

## VI. ACKNOWLEDGEMENTS

Finally, we would like to thank Dr. Malek ALI for his help that facilitates our work and that allows us to begin this research in the Albaath University.

x	xit`a:b	letter	خطاب
G	Garb	west	غرب
X\	X\ilm	dream	حلم
?` (?\)	?`alam	flag	علم

## APPENDIX A: ARABIC ALPHABET FROM SAMPA

Symbol	Keyword	English gloss	Orthography
--------	---------	---------------	-------------

### Consonants

#### Plosives

b	ba:b	door	باب
t	tis?`	nine	تسع
d	da:r	home	دار
t`	t`a:bi?`	stamp	طابع
d`	d`arab	he hit	ضرب
k	kabi:r	large	كبير
g	gami:l	beautiful	جميل

(in Egyptian pronunciation)

?`	?akl	food	أكل
q	qalb	heart	قلب
p	paris	Paris	پرس

#### Fricatives

f	fi:l	elephant	فيل
v	nivi:n	Nevien (personal name)	نفين
T	Tala:T	three	ثلاث
D	Dakar	male	ذكر
D`	D`ala:m	darkness	ظلام
s	sa?`i:d	happy	سعيد
z	zami:l	colleague	زميل
s`	s`aGi:r	small	صغير
S	Sams	sun	شمس
Z	Zami:l	beautiful	جميل

Symbol	Keyword	English gloss	Orthography
--------	---------	---------------	-------------

h	hawa:?`	air	هواء
---	---------	-----	------

### Nasals

m	ma:l	money	مال
n	nu:r	light	نور

### Trill

r	rima:l	sand	رمال
---	--------	------	------

### Lateral

l	la:	no	لا
l`	?al`T`ah	God	الله

### Semivowels

w	wa:hid	one	واحد
j	jawm	day	يوم

### Vowels

i	D`il	shadow	ظل
a	X\al	solution	حل
u	?`umr	age	عمر
i:	?`i:d	feast	عيد
a:	ma:l	money	مال

## REFERENCES

- [ 1] VoiceXML specification. Voice browser working group of world wide web consortium. <http://www.w3.org/TR/VoiceXML20>.
- [ 2] <http://www.vxml.org/>.
- [ 3] OpenVXI from CMU at: <http://www.speech.cs.cmu.edu/openvxi/index.html>.
- [ 4] Beasley, R. et al.: Voice Application Development with VoiceXML. USA: Sams Publishing, August 2001. (ISBN 0-672-32138-6).
- [ 5] voice XML programmer's guide 6th edition from IBM.
- [ 6] Maria Moutran Assaf "A Prototype of an Arabic Diphone Speech Synthesizer in Festival".
- [ 7] Husni-AI-Muhtaseb, Moustafa Elshafei and Mansour Al-Ghamdi "TECHNIQUES FOR HIGH QUALITY ARABIC SPEECH SYNTHESIS" College of Computer Science and Engineering, King Fahd University of Petroleum and Minerals, 2003.
- [ 8] MBROLA-project The MBROLA project towards a freely available multilingual speech synthesizer: <http://tcts.fpms.ac.be/synthesis/mbrola.html>
- [ 9] freeTTS from sourceForge <http://freetts.sourceforge.org>.
- [ 10] SAMPA. computer readable phonetic alphabet. <http://www.phon.ucl.ac.uk/home/sampa/home.htm>.
- [ 11] CMU Sphinx: a speaker-independent large vocabulary continuous speech recognizer. It is also a collection of open source tools and resources that allows researchers and developers to build speech recognition systems. <http://www.speech.cs.cmu.edu/>; <http://cmusphinx.sourceforge.net/html/cmusphinx.php>
- [ 12] M. M. El Choubassi, H. E. El Khoury, C. E. Jabra Alagha, J. A. Skaf and M. A. Al-Alaoui "Arabic Speech Recognition Using Recurrent Neural Networks".
- [ 13] THE IBM 2006 GALE ARABIC ASR SYSTEM.
- [ 14] Ramzi A. Haraty and Omar El Ariss Lebanese American University, Beirut, Lebanon: "CASRA+: A Colloquial Arabic Speech Recognition Application".
- [ 15] D. Vergyri, K. Kirchhoff, R. Gadde, A. Stolcke, and J. Zheng, "Development of a conversational telephone speech recognizer for levantine arabic," in Interspeech-2005, 2005.
- [ 16] Kirchhoff, K., et al., 2002. Novel approaches to Arabic speech recognition. Final Report from the JHU Summer Workshop, Tech. Rep., John Hopkins University.
- [ 17] A.Messaoudi, Gauvain J.-L., and L. Lamel, "Arabic broadcast news transcription using a one million word vocalized vocabulary," in ICASSP-2006, 2006.
- [ 18] M. Elshafei Ahmed, "Toward an Arabic Text-To-Speech System", The Arabian Journal for Science and Engineering, Volume 16, Number 4, October 1991.
- [ 19] H. Satori, M. Harti, and N. Chenfour: "Introduction to Arabic Speech Recognition Using CMUSphinx System"