

# Text To Speech Conversion using Lab view

S.Sundaramahalingam, S.Arockiaraj, N.Vanaja, S.Alagammal

**Abstract:** There is about 45 million blind people and 135 million visually impaired people living across the world. Having the Disability of visual text reading has had a massive influence on the quality of life for the visually disabled people. Even though there have been several devices discovered and designed specifically to support the visually challenged people. By aiding them in viewing objects using an alternating sense such as sound and touch, the progress of text reading device is still in its development phase. Text to speech synthesizers (TTS) have already been implemented using MATLAB and OCR software. In this paper we study the possibility of transforming the text input into speech (text-to-speech) utilising the Lab VIEW's programming environment. In order to achieve this purpose, the text-to-speech interfaces already provided by the Microsoft Speech SDK for TTS applications have been studied. An user friendly application for our VI is developed from scratch and it can be installed effortlessly in any computer system without any cost for installation. As engineer's main aspiration lies in making inventions that makes life simpler with the help of technologies, this is one such invention that aid to simplify real life difficulties.

**Index Terms:** Microsoft speech SDK, Labview, Text to Speech Synthesizers

## I. INTRODUCTION

The issue faced with the speech synthesis domain is quite extensive. Still there quite a few problems that are wide spread, existing with the text at the pre-processing stage, as the conversion of the numerals, abbreviations, and acronym to speech synthesis is still an headache as the viable solution for these issues is still in its primitive stage. Written text contains no explicit emotions and pronunciation of proper and foreign names is sometimes very anomalous. Laboratory virtual instrument engineering workbench, is full form of LabVIEW[1], it is a platform where can we design systems and this environment is used as a visual programming language space provided by the 'National instruments'. One of the main functions of using LabVIEW platform used for the data acquisition, instrument control and industrial automation. The main function of the text-to-speech processor is to be involved in the conversion process of converting normal language text into the speech. The synthesized speech can be

formed by concatenating the pieces of recorded speech that is stored in the database. TTS enables the access of your content to a larger set of population, such as those with literacy difficulties, learning disabilities, reduced vision and those learning a newer language [2]. It also opens way for anyone who is looking for easier ways to access digital content. MICROSOFT SDK SPEECH LIBRARY 5.1 is an Application Programming Interface established by Microsoft to enable speech synthesis within Windows applications. Significant technologies such as text to speech can have had a prolonged and larger impact in people's lives by making it simpler and accessible for individuals, organizations, institutions and businesses is essential in taking this product reach a larger set of audience by assisting them in their daily activities and enabling them to achieve higher.

## II. METHODOLOGY

In this paper, we have proposed to analyse the possibilities that are involved in the conversion process of the written language into speech (text-to-speech) using the LabVIEW programming domain. In order to achieve this objective, we initiated our work by inspecting the text-to-speech interfaces provided by the Microsoft Speech SDK for TTS applications as shown in figure 1.1.

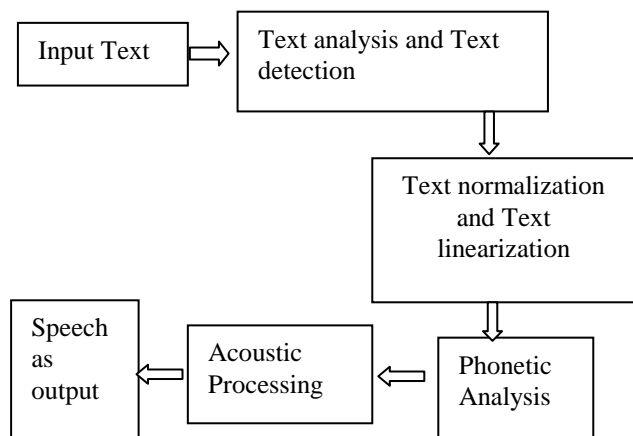


Fig 1 Block diagram

## BLOCK DIARAM DESCRIPTION

### A. Text Analysis & Detection:

The overall structure of the text to speech conversion process starts with the Text Analysis section, which is part of preprocessing. At first it analyzes the input text and then organizes into manageable list of words. After organizing them into a list, it transforms them into full text.

Revised Manuscript Received on 22 May 2019.

\* Correspondence Author

**S.Sundaramahalingam\***, Department of Electrical and Electronics Engineering, Mepco Schlenk Engineering College, Sivakasi, India.

**S.Arockiaraj**, Department of Electrical and Electronics Engineering, Mepco Schlenk Engineering College, Sivakasi, India.

**N.Vanaja**, Department of Electrical and Electronics Engineering, Mepco Schlenk Engineering College, Sivakasi, India.

**S.Alagammal**, Department of Electrical and Electronics Engineering, Mepco Schlenk Engineering College, Sivakasi, India.

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC-BY-NC-ND license <http://creativecommons.org/licenses/by-nc-nd/4.0/>

Text detection localizes the text areas from printed documents.

## B. Text Normalization & Linearization:

In the Text Normalization process, the conversion of the text to pronounceable form takes place. The main objective in running this process is to identify punctuation marks and pauses that exists between the words. Previously the text normalization process was predominantly used for converting all letters of lowercase or upper case, to remove punctuations, accent marks , stop words or “too common words” and other diacritics from letters.

## C. Phonetic Analysis:

It provides phonetic alphabets. The main need for the Phonetic Analysis is to make the conversion of grapheme to phoneme is done. This process also involves the conversion of orthographical symbols into phonological symbols.

## D. Phonetics and Theory of Speech Production:

There is an array of concepts and terminologies that is involved with the Speech processing and language technology carries. To comprehend how contrasting the speech synthesis and analysis methods work then one must have adequate knowledge over the speech production, articulatory phonetics, and other associated terminologies.

## III. LABVIEW PROGRAMMING

Laboratory virtual instrument engineering workbench is the extended expansion for the acronym LabVIEW. This is mainly used as a vital system-design platform. With the environment provided is especially used as a visual programming language platform that is exclusively designed by ‘National instruments’. It is principally utilized for functions such as data acquisition, instrument control and industrial automation on diverse platforms. The programming language used by LabVIEW application is Dataflow programming. The execution and the output flow is completely dependent on the graphical block structure diagram (LabVIEW) source code on which the programmer connects the diverse function-codes by using the wires and connecting them. One of the main functions of these wires is that it is used to send or transmit values of the variables and when all the input values has been obtained then the nodes can perform its function by executing instantly[3-4]. In other case, there is a possibility of parallel execution when there is multiple nodes utilised simultaneously.

The LabVIEW is the main application that binds the design of user interface (called front panels) in a way that makes it part of the development cycle. The program or the routines that are carried out in the LabVIEW are known as *virtual instruments(VI)*. The elements that are kept in each block has three components namely a front panel, a block diagram and a connector panel. The construction of the front panel is predominately done by utilising the constituents like indicators and controls. The Controls act as the inputs to the system and they allow information to flow from the user to the VI by posing as a supplier. The output for the system is given through the Indicators that is present in the system, and the output result display is completely based upon the inputs that are given to the VI. The VI that is present in block diagram of

the other calling VI is represented through the connector panel is available. The back panel also known as the block diagram is where the graphical source code gets stored. The objects that have been placed in the front panel will appear in the back panel as terminals. In addition to this the back panel also contains the structures and functions that are solely responsible for performing control operations and supply the required data. The Nodes are interconnected with each other through the wires. The VI can either run as a program with front panel as user interface. One of the main advantages of using this software is that the graphical approach which permits the user with non-programming background build program with ease as they can drag and drop functions that are needed in accordance to their system requirements. The representation is as such of the lab equipment that they are already familiar with, so with ease they can build programs[5-6].

## A. Interfacing to devices:

LabVIEW does back the interfacing of devices, instruments, cameras etc. The LabVIEW users are also allowed to use their hardware by linking them in either of the two ways namely, by typing out the direct bus command or by using high level device specific drivers that has the native LabVIEW function nodes that can monitor the device.

## B. Overview of ActiveX

ActiveX is the general name that represents a set of Microsoft Technologies that has the capabilities to enable the code that allows you to reuse code and interconnect individual programs concurrently that is particularly designed to satisfy and suit your computing requirement. Under the basis of COM (Component Object Model) technologies, ActiveX is an outright addition to the previously built technology called OLE (Object Linking and Embedding). The program utilized does not regenerate components, in this they reuse the defined components, which gives user the complete power to concatenate applications together. The LabVIEW also provides its assistance for the ActiveX automation by acting as a server in addition to this it also supports ActiveX Container, and ActiveX Event

**Table 1 ActiveX palette**

Palette Object	Description
Automation Open	Returns an automation reference number that indicates a particular ActiveX object.
Close Reference	Closes a reference number that is in association with an open VI, VI object, and an open application instance.
Invoke Node (ActiveX)	Invokes a method or action on a reference. Most methods have associated parameters.
Property Node (ActiveX)	Enabling the property node in an ActiveX environment, has two functions like Gets (reads) or sets (writes) properties of the reference that is available in the palette.

Register Event Call back	Registers is a VI that is to be called in an occurrence of an event. This function is mainly used to register and handle .NET and ActiveX events that occur. LabVIEW in general operates this type of the input reference wired to each and every item to regulate the events that you can register for.
Static VI Reference	This function in the palette maintains a static reference to a VI. You are even allowed to configure the Static VI Reference function to generate a output with the generic or strictly typed VI reference. After placing the Static VI Reference function on a block diagram, double-click on the function that will display a file dialog box that leads you to select a VI.
To Variant	This supports us in the conversion of any LabVIEW data into a variant data. There is an alternative way also to use this function to convert ActiveX data to variant data.
Unregister For Events	This function is used in order to unregister all events associated with an event registration reference number.
Variant To Data.	This function is an invert function of the To Variant function that is utilized in Conversion of variant data to a LabVIEW data type so that LabVIEW can display or process the data.

### BLOCK 1-

This Block consist of speech lib.1speech voice2

### BLOCK 2-

The Block 2 has the Automation open Block

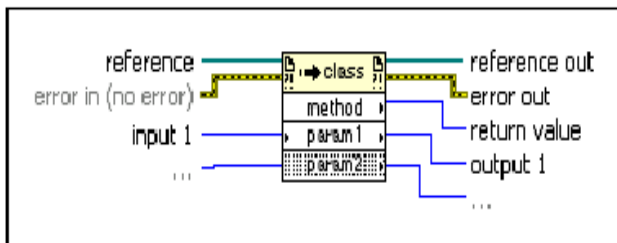


Fig 2 Automation open

It returns an automation Reference number, that point to a particular ActiveX object.

- **Automation Refnum** It gives out the type of object that can be assimilated for the output through the **Automation Refnum** output purpose.
- Machine name is used for the indication that in which machine the VI should access in order to open the **Automation Refnum**. If the output has no machine name in it, then the object can be accessed through the local machine.
- If the condition **open new instance** is TRUE, then the LabVIEW generates a new occurrence of the Automation Refnum. If the condition is FALSE which is set as the default condition, then the LabVIEW would try to connect an instance of occurrence of the refnum that is already existing open in the dialog panel. If the current attempt is ineffective and leads to the failure then the LabVIEW would create and opens up a new instance as it response over the unsuccessful attempt.
- **Error in** function is used in the illustration of the error conditions that has occurred right before this node runs. The

need of this functionality is to provide standard error by utilizing the input function.

- **Automation Refnum** is the refnum function that is in close association with the currently available ActiveX object.
- **Error out** this function only contains error information. This output from this functionality provides us with the standard error out if there is an error.

### BLOCK 3-

Invoke node

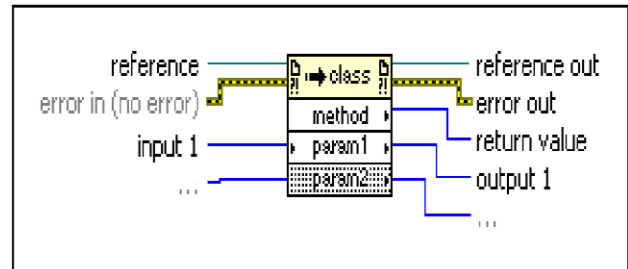


Fig 3 Invoke node

The main function of this node is to provide a quick response, if the reference is taken. If the node is changed on the basis of the VI Server Application Class (or) VI Instrument Class and In case the reference goes unwired this will lead the reference to default and then default to the on-going Application or VI. The LabVIEW application contains Invoke Nodes that are already configured to have the access to XML methods and ActiveX methods.

- **Reference** is the refnum that is in close association with the object, which you want to call on a method or execute an action that is a bit tougher than ordinary. Whether the invoke node class is either an application or VI, there is no need to use a refnum to this very input. The prevailing application instance is the default for the application class that is under consideration. The VI accommodating the Invoke Node is the default set up in the VI class.
- **Input 1...n** are the sample inputs that are available in the parameters of this approach.
- **Return value** is the sample return value that exists in this approach.

### BLOCK 4- Property node

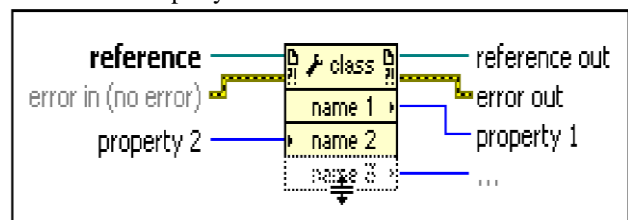


Fig 4 Property node

This Node's actual function lies in the reading or writing over the features that are present in the reference that is available. Actually the Property Node for all instances conditions itself with the object classes that you have in the **reference**.

- **reference** is the refnum that is in close association with the object that is selected and the properties can be set or get in accordance based on the user preference.



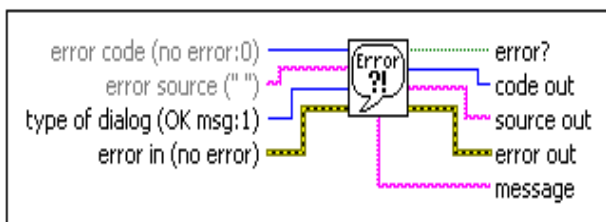
In general the Property Node class can be classified as an Application or VI, there is no need in wiring a refnum to the viable input given through this node. As said before, the Application class has the default current application instance. But there are minor changes when it comes to the VI class, in the default condition, the VI holds the Property Node. In this we can also wire a LabVIEW class to the **reference** input in order to have the private data of the LabVIEW which cannot be accessed by the public class.

- **error in** function is used in the property node to provide accurate description of the error conditions that has taken place prior to the functioning of the node. The standard error is provided through this input by utilizing this functionality.
- **Property 2.n** are the samples of features that are available to be set (write) by the user.
- **Reference out** This mainly used in returning the **reference** without any variation.
- **Error out** This primarily holds the fundamental information about the errors that are present currently. The output provided through this function has the purpose of providing standard error out.
- **Properties 1.n** are the samples of features that are available to be get (read) by the user.

### BLOCK 5- Announcement

This Block provides us with an space to write the text with words that is highly important and in relation with the current activities that is going to the conversion process to be converted to speech.

### BLOCK 6- Error out



**Fig 5 Error handler**

This is the cluster that usually contains this three elements:

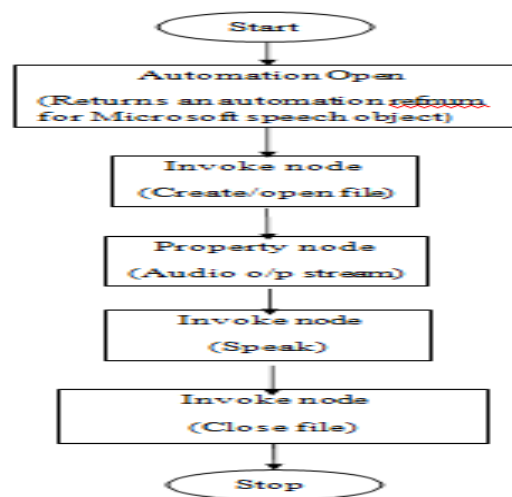
- 1-Status (Boolean TRUE or FALSE)
- 2-Code (long 32bit integer)
- 3-Source (string format)

### Cluster Functions

The cluster functions is predominantly utilized for the Usage of the Cluster, Class, and Variant VIs with the purpose to innovate and improve the clusters. This can be illustrated with an example, where you can perform duties resembling to the following tasks:

- Where you can extract the individual data elements from a given cluster.
- Given the possibility to add the independent data of the elements to a cluster.
  - Ability to break the cluster off by breaking it to make each of its individual data elements available.

### C. Programming flow of ACTIVEX used in LabVIEW



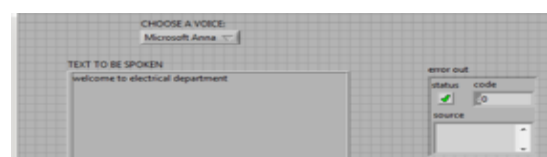
**Fig 6 Flow chart**

The ACTIVE X sub pallet that is available in the Communication pallet is usually utilized for the sending and receiving of the data between each applications that is currently in use. This provides a model for the inter application communication that are using multiple programming language as source of a variant type which has the ability to be implemented in dissimilar platforms. The *Microsoft Speech Object Library that is of the Version 5.1* has been basically utilized in building speech-enabled apps that has the potential to retrieve the voice and large audio output files containing information from the computer. In the function present in the library permits us to select the voice and the audio device in which you want to hear the prepared text to be read. You can also vary the speed rate of the text that is converted to speech and also volume of the voice heard by us.

### D. FRONT PANEL OF TTS PROCESSOR

The Controls act as the inputs to the system and they allow information to flow from the user to the VI by posing as a supplier. The output for the system is given through the Indicators that are present in the system, and the output result displayed is completely based upon the inputs that are given to the VI. The objects that have been placed in the front panel will appear in the back panel as terminals. In addition to this the back panel also contains the structures and functions that are solely responsible for performing control operations and supply the required data.

### E. OUTPUT OF LabVIEW BASED TTS PROCESSOR



**Fig 7 Front Panel**

#### IV. APPLICATIONS

There is a wider range of application for the TTS processor. In the near future it will become a boon for the society. As it can be used by the disabled people living all over the world and can make them similar to normal people i.e. removing the disability barrier for the people facing complex in the society. Some of the major applications are mentioned below:

- For the **Dyslexic** people, facing the problem in reading the text, TTS converter can be installed anywhere i.e. either in their systems or their mobile phones.
- It is very much useful for the **dumb** peoples. As they face severe problems in making the other person who is willing to help him understand their need. So they can type easily and in the announcement window of TTS processor and the speech sound will make other's understand their need.
- It can be used to teach the **pre-illiterate** children. As it can teach them how to pronounce the words in their basic stage of learning..

#### V. CONCLUSION

There are many complex hardware circuits for text-to speech conversion, already available in the present market. They face a lot of difficulty in fabricating them as IC and they are facing lots of problems in signal processing part. So in the world of great competition between hardware and software, the software always wins in the cases of compatibility and the user friendliness. As they can be easily modified and updated. Hence this LabVIEW based Text-to speech processor can be easily installed in the mobile phones and the other compatible devices. The application developed is user friendly, cost effective and gives the result in the real time. Moreover, the program has the required flexibility to be modified easily if the need arises and it can be updated within minutes with the internet connection available at close proximities.

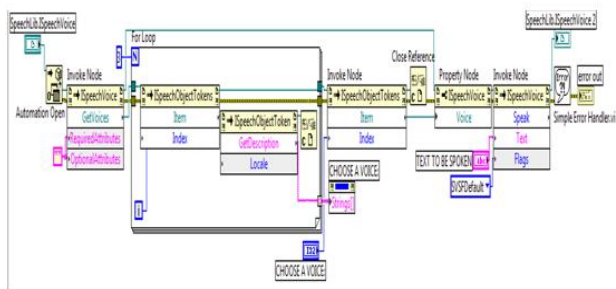


Fig 8 Block Diagram

#### ACKNOWLEDGMENT

The authors would like to gratefully acknowledge the technical support from the Mepco Schlenk Engineering College, Sivakasi

#### REFERENCES

1. L.R. Rabiner, B.-H. Juang, 1993 "Fundamentals Of Speech Recognition" (Prentice-Hall, Englewood Cliff).
2. Olabe, J. C.; Santos, A.; Martinez, R.; Munoz, E.; Martinez, M.; Quilis, A.; Bernstein, J., "Real-Time Text-To-Speech Conversion System For Spanish," Acoustics, Speech, And Signal Processing, Ieee

- International Conference On Iccasp '84. , Vol.9, No., Pp.85,87, Mar 1984.
3. A Novel Model For Speech To Text Conversion, Deepav. Jose, Alfateh Mustafa, Sharan R, Published In International Refereed Journal Of Engineering And Science (Ijres) Issn (Online) 2319-183x, (Print) 2319- 1821 volume 3, Issue 1 (January 2014), Pp. 39-41.
4. T. Dutoit, (1997). (An Introduction To Text-To-Speech Synthesis. Kluwer Academic Publishers, Dordrech/ Boston/London
5. H. Hon, A. Acero, X. Huang, J. Liu, M. Plümpe. "Automatic Generation Of Synthesis Units For Trainable Text-To-Speech Systems," Proceedings Of Iccasp 98 (Cd-Rom), 1998.
1. 6. Decadt, Bart; Duchateau, Jacques; Daelemans, Walter; Wambacq, P., "Phoneme-To-Grapheme Conversion For Out-Of Vocabulary Words In Large Vocabulary Speech Recognition," Automatic Speech Recognition And Understanding, 2001. Asru '01. Ieee Workshop On, Vol., No., Pp.413, 416, 2001.

#### AUTHORS PROFILE



**Mr. S. Sundara Mahalingam** graduated in Electrical and Electronics Engineering from SSN College of Engineering, Chennai, Tamilnadu, India in 2010. In 2012, he received Master of Engineering (M.E) degree in High Voltage Engineering from the College of Engineering Guindy, Anna University, India. He had undergone project internship training at Crompton Greaves LTD, Mumbai. He had 6 years of teaching experience and currently he is working as an Assistant Professor in the Department of Electrical Engineering at Mepco Schlenk Engineering, Sivakasi, Tamil Nadu, India.. Now, he is pursuing his research in the field of High Voltage Engineering under Anna University, Chennai, India. His research activities are focused on FEM, Nano composite magnetic materials, Nano composite dielectric materials, and Smart Grid. He has published many international journals.



**Mr. S. Arockiaraj** graduated in Electrical and Electronics Engineering from Francis Xavier Engineering College, Tamilnadu, India in 2009. In 2013, he received Master of Engineering (M.E) degree in Power System Engineering from the College of Engineering Guindy, Anna University, India. He had 6 years of teaching experience and currently he is working as an Assistant Professor in the Department of Electrical Engineering at Mepco Schlenk Engineering, Sivakasi, Tamil Nadu, India. He has published many international journals.



**Mrs. N. Vanaja** graduated in Electrical and Electronics Engineering in 2007 and she received Master of Engineering in Power Electronics and Drives from Mepco Schlenk Engineering College, Sivakasi, Tamil Nadu, India, in 2009. She had 1 year of teaching experience in PSNA College of Engineering and Technology, Dindugal. From 2010 to 2015, she was an Assistant Professor in Electrical Engineering Department at Mepco Schlenk Engineering, Sivakasi. Since 2016, she has been an Assistant Professor (Sr. grade) in the same college. Now, she is pursuing her research in the domain of Optimization of Renewable Energy Sources Integration to Smart Grid under Anna University, Chennai, India. Her research interests are on Power system stability, Artificial intelligence techniques and Smart Grid.



**Mrs. S. Alagammal** graduated in Electrical and Electronics Engineering from Thiagarajar College of Engineering, Madurai, Tamil Nadu, India, in 1995. In 2008, she received Master of Engineering (M.E) in Power Electronics and Drives from College of Engineering, Guindy, Anna University (AU), Chennai, India. She had 8 years of teaching experience in Adhiparasakthi Engineering College, Melmaruvathur. From 2009 to 2015, she was an Assistant Professor (AP) in Electrical Engineering Department at Mepco Schlenk Engineering, Sivakasi, Tamil Nadu, and India. Since 2016, she has been an AP (Sr. grade) in the same college. Now, she is pursuing her research in the domain of power electronics applications in solar energy under AU, Chennai, India. Her research interests are on applications of power electronics converters in PV systems, MPPT controller using artificial intelligence techniques.