A GENERAL PURPOSE SPEECH EDITOR, THE SPEAK LANGUAGE.

H.S.J.Purvis.
Speech Hearing & Language Research Centre
Macquarie University

ABSTRACT- This paper describes a simple computer language known as the speak language that is used in a general purpose speech editor to output words or sentences in a defined sequence with specified timing.

INTRODUCTION

The speech editor is used to record lists of words or sentences in a given sequence and timing. It was decided that the most versatile way of doing this would be to use a simple computer language. The advantage of this method is that it imposes fewer limitations and permits the more complex tasks to be performed easily. The language can be extended as required to meet future needs. This would be difficult if a less flexible method were used. The disadvantage is that A program must be written and this makes the simpler applications more difficult than they might have been if some other method was used.

IDENTIFYING THE SOUNDS

The speech editor is used to divide a speech file into sections that contain words, sentences etc., each section has a beginning and end marker.

A set of these sections is known as a level, and is given a letter from 'A' to 'Z'. Level 'A' consists of one section whose beginning mark is at the start of the file and end mark is at the end of the file. Levels 'B' to 'Z' consist of a number of sections, numbered from 1 to a maximum of 10,000. There is no relationship required between sections at different levels, a level 'C' section may be within a level 'B' section, they may overlap, or they may be unrelated.

Some speech editor commands operate on all sections with a given level, for example the command to adjust the RMS levels of sections to a common value.

The speech editor provides for one or two speech channels, so a channel number is required to complete the identification, the channels are numbered 1 and 2.

The identification of a sound consists of four parts, the data set name, the level, the number, and the channel.

Data_Set = Jilly Level = B Number = 1 Channel = 1

The term data set refers to the four files that contain the speech and related data. They are the header file containing the fixed information such as sample rates, block sizes, titles etc. the wave file that contains the speech data, the computed data file that contains derived data such as the intensity curve or the pitch curve, the description file that contains the marks used to divide the speech data into sections

THE PROGRAM

The program is written using a text editor. It consists of one to three sections, the variables section, the definitions section, and the procedures section. The procedures section is required but the other two sections are optional. Here is an example of a program that contains all three sections :-

```
Variables
{ All variables are 4 byte integers and MUST be defined. }
{ all statements must end with a semicolon ';' }
Count  Number;
Definitions
{Sound definitions enable a sound to be referred to in a speak command  by its NUMBER or by the use of a variable. }
Define  Data_Set = Jilly Level = B Number = 1 Channel = 1;  { number 1 }
Define  Data_Set = Jilly Level = B Number = 2:4 Channel = 1; { numbers 2 to 4 }
Procedures
Procedure Part_1
[
        Speak Left = 3          Spacing = 5.0;
        Speak Left = Number     Spacing = 1.0;
        Speak Left = 2          Spacing = 1.0;
]
{ In every program there MUST be a procedure called Main. }
Procedure Main
[
        Left_Data_Set           := Jilly; { These 4 variables are built in }
        Left_Level              := A;
        Left_Number             := 1;
        Left_Channel            := 1;

        Count := 3;             { Assignment }
        Number := 0;

        Repeat Count Times
        [
                speak Left = 1                  Spacing = 0;
                Inc Number;                     { Number := Number + 1  }
                Part_1;         { call procedure Part_1 }
        ]
        Speak Left = (Data_Set = Jilly Level = B Number = 1 Channel = 1) Spacing = 5.0;
]
        End
```

Comments are enclosed in curly brackets {....} these are ignored by the speech editor.

THE VARIABLES SECTION

The variables section is used to name the variables that are used in the program. If no variables are used then this section can be omitted. Up to 100 variables may be used in a program, each variable name consists of up to 30 characters, the letters, digits, and the '_' may be used.

There are ten variables that are built in, these can be used without naming them in the variables section, they are :-

LEFT_DATA_SET, LEFT_LEVEL, LEFT_NUMBER, LEFT_CHANNEL
RIGHT_DATA_SET, RIGHT_LEVEL, RIGHT_NUMBER, RIGHT_CHANNEL
SPACING, SILENCE

If any of the four parts of a sound identification are omitted then the value will be taken from the corresponding built in variable.

THE DEFINITION SECTION

The definition section is used to define a sound and give it a number. The sounds are given numbers corresponding to their order in the definition section. Up to 500 sounds may be defined. If no sounds are defined the definition section may be omited.

```
Define Data_Set = Jilly Level = B Number = 10 Channel = 1; { number 1 }

Define Data_Set = Jilly Level = B Number = 6:8 Channel = 1; { numbers 2 to 4 }
```

The first definition defines sound number one. The second definition shows how a sequence of sounds may be defined using one definition statement, in this case sounds two, three, and four are defined. When a sound has been defined as a number, this number can be used to refer to the sound in the procedures section. Note that this number should not be confused with the Number that forms part of the sound identification.

THE PROCEDURES SECTION

A program in the speak language must contain a procedures section or it will not do anything. It may have of up to twenty procedures but it must contain a procedure called MAIN and this must be the last procedure.

```
Procedure Part_1
[
        Speak Left = 3          Spacing = 5.0;
        Speak Left = Number     Spacing = 1.0;
        Speak Left = 2          Spacing = 1.0;
]
```

A procedure consists of the word procedure followed by the procedure name, the name may consist of up to 30 characters, the letters, digits, and '_' may be used. The body of the procedure is enclosed in square brackets.

Program execution begins with the procedure MAIN and stops when this procedure is finished. The example above shows the commands at present available in the speak language.

The assignment statement

```
Count := 3;
```

The assignment statement is limited to assigning a constant or the value of a variable to a variable. The speak language is not intended for mathematics but this statement could be extended in the future as required.

The repeat statement

```
Repeat Count Times
[
        speak Left=1 Spacing=0;
        Inc Number;                    { Number := Number + 1 }
        Part_1;                        { call procedure Part_1 }
]
```

The repeat statement may use a variable or a constant to determine the number of repeats. The statements to be repeated are enclosed within square brackets.

The inc statement.

```
Inc Number;                            { Number := Number + 1 }
```

The inc statement is used to add one to a variable.

The procedure call

```
Part_1;                                { call procedure Part_1 }
```

A procedure is called by placing its name in another procedure.

The speak statement

There are two forms of the speak statement.

```
speak Left=1 Spacing=1.0;
```

This form is used to speak a sound that has been defined in the definition section. In this example the first sound in the definition section is sent to the left sound channel, the total time of the silence before the sound and the sound is one second as set by the 'spacing = 1.0' section of the statement. If the sound is longer than one second then no silent period will be used. This method is used to produce a set of sounds evenly spaced even though they vary in length. If the length of the silence between the sounds must be constant then use the word 'delay' instead of 'spacing'.

```
Speak Left=(Data_Set=Jilly Level=B Number=1 Channel=1) Spacing=5.0;
```

This form is used to speak a sound that has not been defined in the definition section. All the information necessary to specify the sound is placed in the speak statement.

```
Speak  Left=(Data_Set=Jilly Level=B Number=1 Channel=1) Spacing=5.0
```

```
Right = (Data_Set = Jilly Level = B Number = 2 Channel = 2);
```

If two channels of sound are used then this form of the speak statement may be used to send sound out of the left and right channels of the sound system. Note that the output channels used do not have to be the same as the channels used when recording the data. Both sounds will start at the same time, so if they are not the same length the will be a period of silence in the channel with the shortest sound.

```
Speak  Left = (Data_Set = Jilly Level = B Number = 1 Channel = 1)
                                        Spacing = 15.0 Delay = 0.5
       Right = (Data_Set = Jilly Level = B Channel = 2);
```

In this speak statement the sound in the left channel will be delayed by half a second after the sound in the right channel. The total time from the start of the silence before the sounds to the end of the sound that finishes last will be 15 seconds.Note that in the right channel, the number has been omitted, this will cause the value of the variable RIGHT_NUMBER to be used.If any of these values are omitted then the value of the corresponding built in variable will be used.

The speak language is able to be extended to meet future needs. One possible extension would be to enable interactive listening tests where the subject presses one of a number of buttons in response to hearing the sound.

EXECUTING A PROGRAM

The command 'SET PROGRAM NAME.SPEAK' causes the speech editor to read the file 'NAME.SPEAK' and translate the program into a number of arrays in memory. The program is checked for erors and messages displayed if any are found.

The command 'SPEAK' causes the program in memory to be executed. The program can be stopped at any time by pressing a key on the keyboard.