

The WireLess Studio voiXtreme voice system

Description

The voiXtreme voice system allows the PDA to say texts and to perform ASR inputs on a WireLess Studio application.

The TTS and the ASR engine are provided by the voiXtreme library, which is installed into the PDA terminal.

Voice engines are separated by language EN (English), FR (French) and ES (Spanish). They are available for download from SofToGo website:

<http://www.softogo.com/com/en/download/Studio.html>

Only one voice engine may be installed in the PDA (en, fr or es).

TTS System

Text data to be said (by TTS) may be supplied by:

- RFPrint function with the flag WLSPELL or WLSPEECH
- RFSay function.

See description of functions in the WSDG manual:

http://www.sof2go.net/man/wst/en/wsdg/RFIO_RFSay.htm

http://www.sof2go.net/man/wst/en/wsdg/RFIO_RFPrint.htm

ASR System

Voice input from user. The ASR system recognizes the user's voice, and converts it into text data using “grammars”. Data input is processed by local application, and then sent to host following programming configuration.

Grammars are recognition algorithms designed to produce text inputs.

Available grammars are:

- “Digits”, will accept decimal digits “0” to “9”, “Cancel” and “Repeat” orders. *Cancel* and *Repeat* keywords are processed by voice library, and transmission to host is performed following input algorithm.
- Commands, will accept “OK”, “Cancel” and “Repeat” orders. *OK*, *Cancel* and *Repeat* keywords are processed by the voice library, and transmission to host is performed following input algorithm.

Keywords change according to the language selected for the ASR.

Voice input is performed by RFInputEx() function. See description of the function in the WSDG manual:

http://www.sof2go.net/man/wst/en/wsdg/RFIO_RFInputEx.htm

Voice transaction (output/input)

The voice transaction is performed by the RFIInputEx() function.

This function allows to perform keyboard and/or voice input, depending on flags used.

The RFIInputEx function allows a TTS “Input Phrase” to be said to user before the ASR recognition process starts.

For the ASR recognition it is mandatory to define the ASR grammar to be used and the ASR algorithm.

Standard voice grammars “*digits*” and “*commands*” are included with the voiXtreme libraries, and custom grammars might be added. Grammar flags are:

WLASR_GRAMMAR_COMMANDS

WLASR_GRAMMAR_DIGIT

WLASR_GRAMMAR_CUSTOM

The ASR Algorithm is controlled by the WLASR_VALIDATE flag, and could be:

- *Single* (default). The engine recognizes the user input, then sends it to host.
- *Validate* (WLASR_VALIDATE). The engine recognizes the user input, repeats it to user and waits for the user to say a command (“*accept*” / “*cancel*”).
 - If *cancel*, it restarts the input from start (voice announce).
 - If *accept*, it sends user input to host.

Input Modes:

The following images describe the different input modes.

Single Voice Input

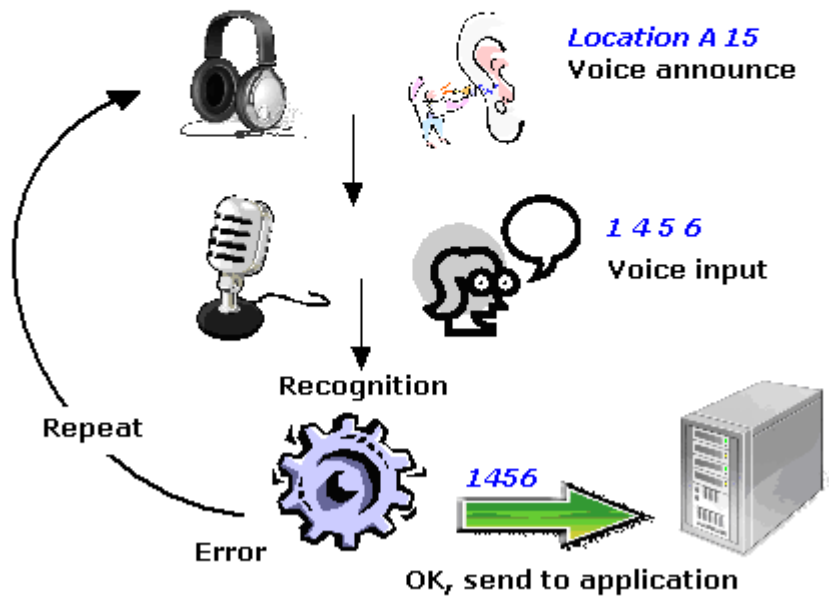


PDA

RFInputEx



User



Single Voice Input Algorithm:

- 1/ RFInputEx (without WLASR_VALIDATE flag).
- 2/ The InputPhrase is said to the user.
- 3/ System waits (or not) end of TTS InputPhrase depending on the value of the field “_ASR/SyncTts” in the Wireless Studio Client configuration.
- 4/ A recognition is started to perform the ASR input, using the grammar indicated by the flags (Digits).
- 5/ End of recognition:
 - Repeat, goes to (2)
 - Error (no recognition, low reliability level) goes to (2)
 - Cancel, sends ESC key to application (LastInputType= WLCOMMANDTYPE)
 - Digits, sends input to the application (LastInputType= ASRTYPE)

Validate Voice Input (input digits)

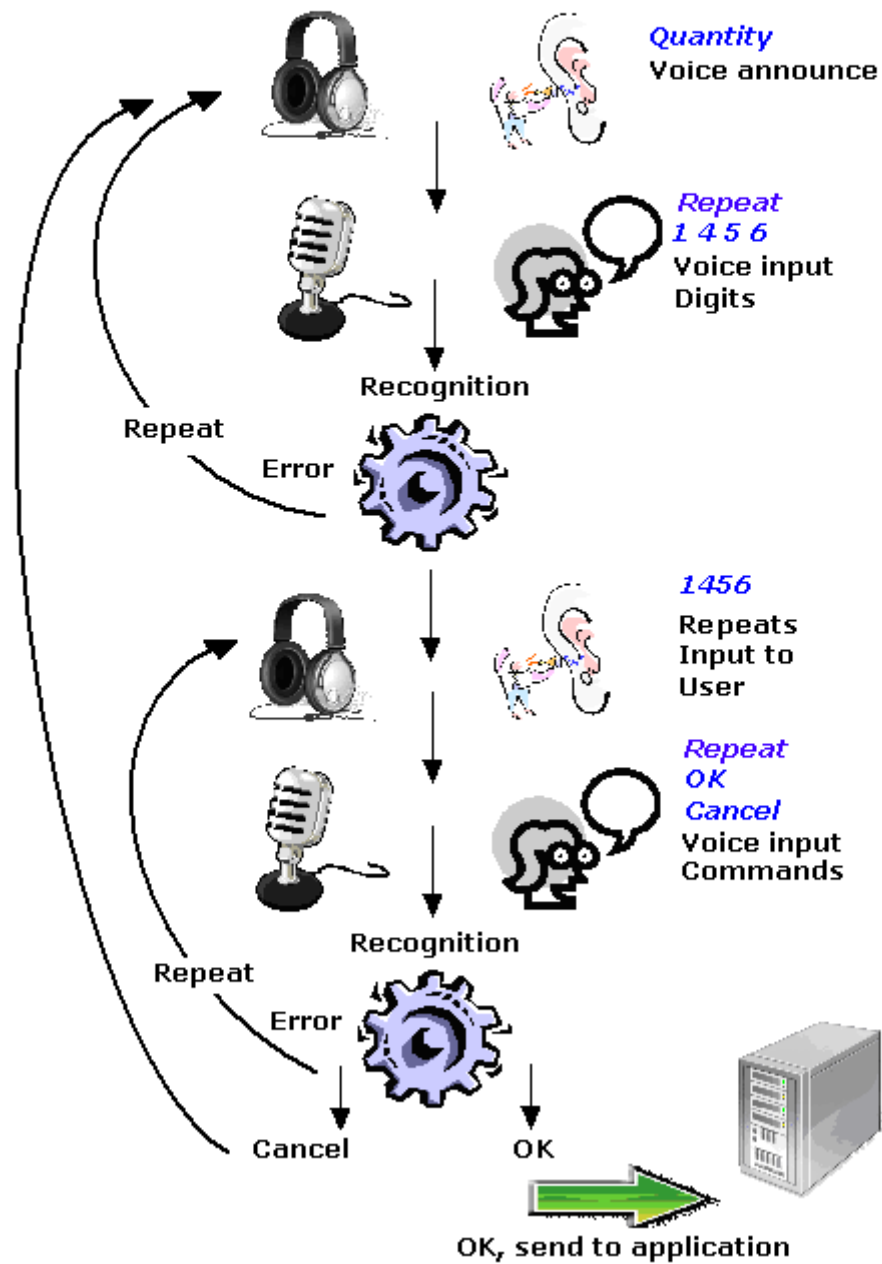


PDA



User

RFInputEx



Validate Voice Input Algorithm (digits):

1/ RFInputEx (with WLASR_VALIDATE flag).

2/ The InputPhrase is said to the user.

3/ System waits (or not) end of TTS InputPhrase depending on the value of the field “_ASR/SyncTts” in the Wireless Studio Client configuration.

4/ A recognition is started to perform the ASR input, using the grammar indicated by the flags (Digits).

5/ End of recognition:

- Repeat, goes to (2)
- Error (no recognition, low reliability level, min / max) goes to (2)
- **Cancel, goes to (9)**
- Digits, goes to (6)

6/ Say the input to the user by TTS.

7/ Perform the ASR recognition, using the “Commands” grammar.

8/ End of recognition:

- Repeat, goes to (6)
- Error (no recognition, low reliability level) goes to (6)
- Cancel, goes to (6)
- Ok, sends to application the input with (LastInputType= ASRTYPE), then exit

9/ Say the **CancelConfirmation** phrase to the user by TTS.

10/ Perform the ASR recognition, using the “Commands” grammar.

11/ End of recognition:

- Repeat, goes to (9)
- Error (no recognition, low reliability level, min / max) goes to (9)
- Cancel, says Cancel to user, goes to (2)
- Ok, sends ESC to application (LastInputType= WLCOMMANDTYPE)

Validate Voice Input (input cancel)

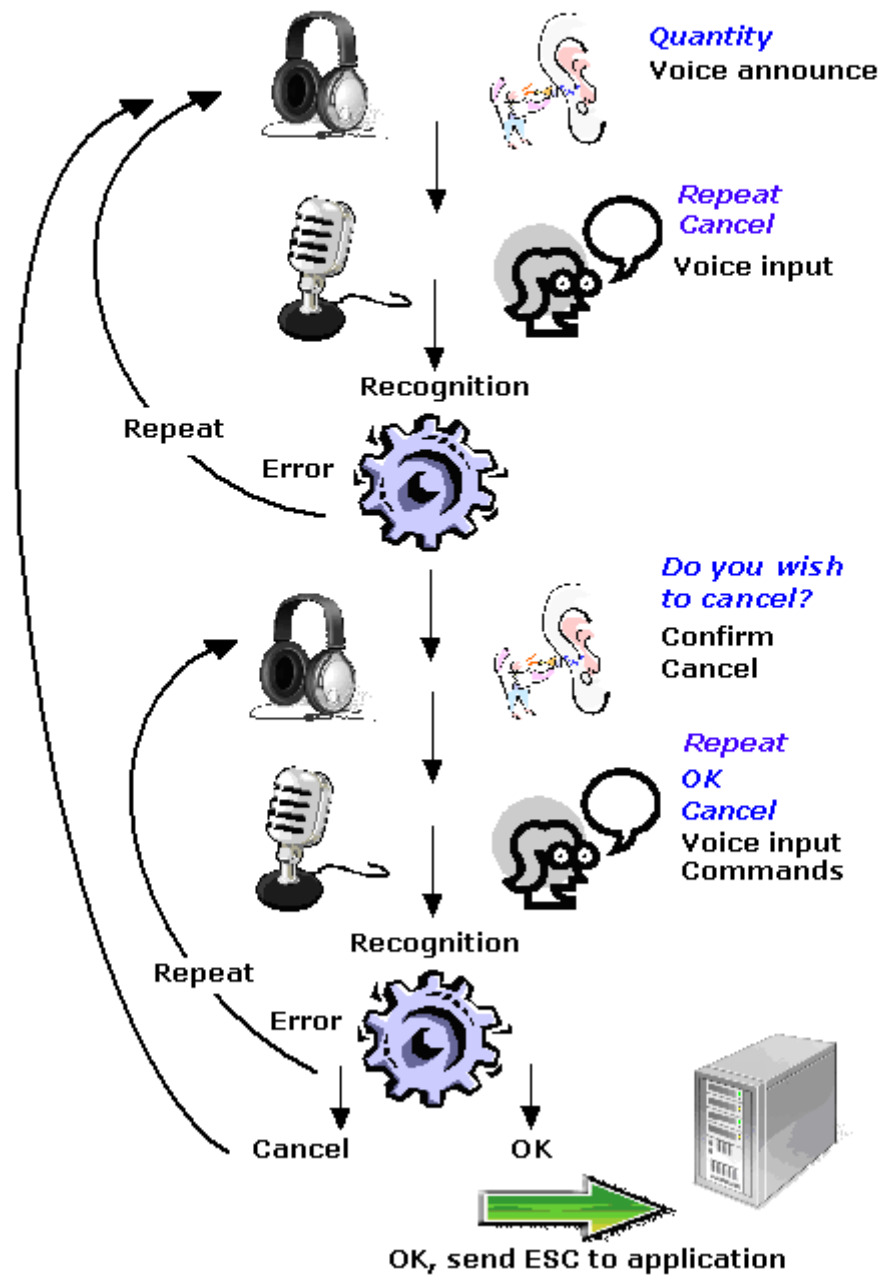


PDA



User

RFInputEx



Validate Voice Input Algorithm (cancel):

1/ RFInputEx (with WLASR_VALIDATE flag).

2/ The InputPhrase is said to the user.

3/ System waits (or not) end of TTS InputPhrase depending on the value of the field “_ASR/SyncTts” in the Wireless Studio Client configuration.

4/ A recognition is started to perform the ASR input, using the grammar indicated by the flags (Digits).

5/ End of recognition:

- Repeat, goes to (2)
- Error (no recognition, low reliability level) goes to (2)
- Cancel, goes to (9)
- Digits, goes to (6)

6/ Say the input to the user by TTS.

7/ Perform the ASR recognition, using the “Commands” grammar.

8/ End of recognition:

- Repeat, goes to (6)
- Error (no recognition, low reliability level, min / max) goes to (6)
- Cancel, goes to (6)
- Ok, sends to application the input with (LastInputType= ASRTYPE), then exit.

9/ Say the CancelConfirmation phrase to the user by TTS.

10/ Perform the ASR recognition, using the “Commands” grammar.

11/ End of recognition:

- Repeat, goes to (9)
- Error (no recognition, low reliability level) goes to (9)
- Cancel, says Cancel to user, goes to (2)
- Ok, sends ESC to application (LastInputType= WLCOMMANDTYPE)