安徽科大讯飞信息科技股份有限公司

ANHUI USTC iFLYTEK CO.,LTD

iFLYTEK Automotive Speech Suite For Multi-platform

安徽科大讯飞信息科技股份有限公司

ANHUI USTC iFLYTEK CO., LTD.

Important statement

Directory

Foreword

Welcome using　iFLYTEK Speech SDK Suite！

The iFLYTEK Speech SDK Suite is a voice sdk which fit for automotive users, it can provide speech recognition, speech synthesizing and other features. Furthermore, it can also provide automotive voice application developers with easy-to-use interface which realize various speech application in car. Main features are:

1) To achieve speech application server based on HTTP protocol, integrated with iFLYTEK latest speech engine which supports speech synthesize, speech dictation, speech recognition etc.;

2) Provide speech client subsystem based on mobile platform, integrated internally with audio processing and Codec. Provide efficient APIs which related to speech synthesize, speech dictation, speech recognition etc.

iFLYTEK is the leading provider of Chinese speech and language technology, serving consumers, businesses and government organizations. Our success is down to having the longest fundamental research time and largest professional research staff, helping us to achieve the best performance in the national research and development “863”evaluation, and the largest capital investment and market share in the domestic speech technology industry.

- iFLYTEK is the only organization specified to evaluate ASR(Automated Speech Recognition) technology by the National “863” Program
- iFLYTEK is the only organization authorized to provide standard Chinese speech recognition databases by the National “863” Program;
- iFLYTEK is in charge of constituting the standard interfaces and specifications for Chinese speech technology
- iFLYTEK has been appointed to draft out the Chinese Information Processing Plan for the National “S863” Program.

The KD2000 text-to-speech system was awarded National Science and Technology progress Award 2nd class- the highest award ever received in the Chinese speech industry.

1. Instruction

1.1. Nouns& Abbreviation

☐ **ISS(Iflytek Speech Suite)**

iFLYTEK Speech Suite--ISS, SDK for automotive speech developers.

☐ **SR (Speech Recognition)**

Speech Recognition—SR. A recognition technology based on speech dictation which focus on recognition of a certain field or special grammatical. Its results are closely related to uploaded contents. The recognition scale can be user-defined after the related command lists or grammar uploaded.

☐ **NLP (Natural Language Processing)**

It is a subfield of artificial intelligence and also is the hardest question in it. It contains the judgment of semantic and can make the computer not only recognize speech but also understand user's purpose.

☐ **TTS (Text to Speech)**

It can convert any text into fluent speech at anytime and anywhere.

☐ **IVW（Iflytek voice wake-Up）**

It is a technology which using speech or audio command word to wake up the stand-by machine without hands.

1.2. Features Introduction

The suite consists of embedded and voice-cloud technologies which provide speech recognition, NLP, speech synthesize, voice wake-up etc. All the features are available on board, off board and hybrid.

1.3. Developing Instruction

1、 This document defined the instruction and structures of iFLYTEK speech recognition, speech synthesize and voice wake-up;

2、 It was made for the those readers who are ready to develop by using the SDK, including product designer, software engineer. The readers can master how to integrate with speech recognition, speech synthesize and voice wake-up after read this document;

3、 Only get our authorization one can use the speech service. Please register on “http://open.voicecloud.cn” to be the voice-cloud developer and apply an APP ID for the your own software;

4、 If the developer has applied multiple APP IDs, different applications need corresponding SDK libs;

5、 For those who has already have the APP ID can use speech recognition and speech synthesize for free. But, it was not allowed to use the APP ID for any semantic request which has described in this document unless the commercial contract was finished already;

6、 Please refer each chapters for specific businesses.

2. Support platform

1、 Support Linux system。

3. Speech recognition

2.1 Interface introduction

2.1.1 List of functions

☐ list of interfaces：

Function Name	Function introduction
ISSSRCreate	Create recognition handle
ISSSRCreateW	Create recognition handle
ISSSRDestroy	Destroy recognition handle
ISSSRStart	Start first recognition
ISSSRAppendAudioData	Input audio record data
ISSSREndAudioData	Stop input record
ISSSRStop	Stop this time recognition
ISSSRUpLoadDict	Compile grammar

3.1.2 Function Introduction

1、 Recognition include grammar recognize and voice dictation；

2、 Recognition support local, internet and local & internet hybrid three model；

3、 Recognition support one way session，not support for multiple way active；

4、 Recognition result was XML from，could parsed by different business。

2.2 Recognition parameter introduction

Null。

2.3 Interface introduction

2.3.1 Create recognition handle

☐ Function prototype

ISSErrID *ISSAPI* ISSSRCreate(HISSSR* phISSSRObj,const char* szResourceDir,pcISSSRInteface pISSSRInteface,void* pUsrArg);

ISSErrID *ISSAPI* ISSSRCreateW (HISSSR*phISSSRObj,const wchar_t* szResourceDir, pcISSSRInteface pISSSRInteface, void* pUsrArg);

☐ **Function introduction**

1、 Create recognition handle；

☐ **Parameter specification**

Param Name	Param Explanation
phISSSRObj [out]	Recognition handle；
szResourceDir [in]	Resource directory, in general：ISSSRCreate version，under windows Chinese enviroment is GBKcoding，under linux environment is UTF-8 coding；ISSSRCreateW version，under windows vc environment is UNICODE-16 coding，under linux gcc environment is UNICODE-32coding。
pISSSRInteface [in]	Recognition callback interface，reference in 识别回调接口声明，can not pass NULL；
pUsrArg [in]	User-defined parameter

☐ **return value**

1、 ISS_SUCCESS：create speech recognition handle success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_FILE_NOT_FOUND：The corresponding resource file not found；
4、 ISS_ERROR_INVALID_PARA：invalid parameter；
5、 ISS_ERROR_MACHINE_CODE_NOT_SET：machine code not set yet;

2.3.2 Destroy recognition handle

☐ Function prototype

ISSErrID *ISSAPI* ISSSRDestroy(HISSSR hISSSRObj);

☐ **Function introduction**

1、 Destroy recognition handle；

☐ **Parameter specification**

Param Name	Param Explanation
hISSSRObj [in]	Recognition handle。

☐ **return value**

1、 ISS_SUCCESS：destroy recognition handle and release resource success；

2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；

3、 ISS_ERROR_INVALID_HANDLE：invalid recognition handle。

2.3.3 Start speech recognition

☐ Function prototype

ISSErrID *ISSAPI* ISSSRStart(HISSSR hISSSRObj,const char* szScene,int iMode,const char* szCmd);

☐ **Function introduction**

1、 Start a specch recognition；

2、 If is recognition state, will stop this time recognition, restart another one；

3、 wheni_Mode = 3，compile command word list will be done in suite wmaintain thread ，will not block ISSSRStart interface。

4、 wheni_Mode = 4， in local semantic, ignore szScene parameters（input text，return semantic result）。

☐ **Parameter specification**

Param Name	Param Explanation
hISSSRObj [in]	Recognition handle
szScene [in]	Specious scene，delimit recognition range
iMode [in]	Recognize model：i_Mode = 0 internet recognition model；i_Mode = 1 local recognition model；i_Mode = 2 local & internet recognition model；i_Mode = 3 local command word recognition；i_Mode = 4 text semantic parse model。
szCmd[in]	i_Mode = 3，command table for local command word，json form, UTF8 coding；i_Mode != 3, not use here

☐ **return value**

1、 ISS_SUCCESS：Start a specch recognition；

2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II。

3、 ISS_ERROR_INVALID_HANDLE：invalid recognition handle；

4、 ISS_ERROR_INVALID_PARA：invalid parameter value，error recognition model or scene；

5、 ISS_ERROR_INVALID_JSON_FMT：invalid Json form；

6、 ISS_ERROR_INVALID_JSON_INFO： necessary grammar message extracted fail from Json。

NB：after ISS_SR_MSG_InitStatusreturn success（ISS_SUCCESS）call ISSSRStart

start a recognition ; after ISSSRStart return ISS_SUCCESS , and call ISSSRAppendAudioData append record 。

2.3.4 Input record data

☐ Function prototype

ISSErrID *ISSAPI* ISSSRAppendAudioData(HISSSR hISSSRObj,short* pSamples,unsigned int nNumberOfToWrite,unsigned int* pNumberOfWritten);

☐ **Function introduction**

1、 Input record data, support input 16Ksample rate、S16-LE、single track PCM record;

☐ **Parameter specification**

Name	Param range
hISSSRObj [in]	Recognition handle
pSamples[in]	Input audio buffer address
nNumberOfToWrite[in]	The number of input samples, un-byte number
pNumberOfWritten[out]	The number of input samples

☐ **return value**

1、 ISS_SUCCESS: input audio record success;
2、 ISSErrID: error code return by speech recognize engine, Refer to appendix II;
3、 ISS_ERROR_INVALID_HANDLE: invalid recognition handle;
4、 ISS_ERROR_INVALID_PARA: invalid parameter value;
5、 ISS_ERROR_INVALID_CALL: error called (ISSSRStart not called, recognition not start);

NB: input audio data must under state of recognition. In order avoid speech recognize engine internal buffer overflow , add pNumberOfWritten parameter。When Client program occurs *pNumberOfWritten !=nNumberOfToWrite, could wait for a while then read-in or abort data of this buffer, Please pay attention to possible during the period of writing data, speech recognition engine callback function

2.3.5 Stop recording

☐ Function prototype

ISSErrID *ISSAPI* ISSSREndAudioData(HISSSR hISSSRObj);

☐ **Function introduction**

1、 Force stop voice recording, obtain recognition result。

☐ **Parameter specification**

Param Name	Param Explanation
hISSSRObj [in]	Recognition handle。

☐ **return value**

1、 ISS_SUCCESS：success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_INVALID_HANDLE：invalid recognition handle；
4、 ISS_ERROR_INVALID_CALL：error called(ISSSRStart not called, recognition not start)；
5、 ISS_ERROR_NO_SPEECH：user's voice record not detected，end recognition，recording need stopped。

2.3.6 Abort recognition

☐ Function prototype

ISSErrID *ISSAPI* ISSSRStop(HISSSR hISSSRObj);

☐ **Function introduction**

1、 Stop the recognition（abort the recognition operation）。

☐ **Parameter specification**

Param Name	Param Explanation
hISSSRObj [in]	Recognition handle。

☐ **return value**

1、 ISS_SUCCESS：stop the recognition success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_INVALID_HANDLE：invalid recognition handle；
4、 ISS_ERROR_INVALID_CALL：error called (ISSSRStart not called, recognition not start)

2.3.7 Compile grammar

☐ Function prototype

ISSErrID *ISSAPI* ISSSRUpLoadDict(HISSSR hISSSR,const char *szList，int bOnlyUploadToCloud);

☐ **Function introduction**

1、 The asynchronous interface compile grammar；
2、 Support songs、singer、contact、apps，grammar compile success or not，measge return by Proc_OnUpLoadDictStatus。
3、 asynchronous interface。After handle created，not need to wait meaasge of ISS_SR_MSG_InitStatus，coulde call ISSSRUpLoadDict upload dictionary。If the last one parameter bOnlyUploadToCloud is 0, then:
every time after call ISSSRUpLoadDict,message return.

ISS_SR_MSG_UpLoadDictToLocalStatus, identify local personalize data upload success or not, return message ISS_SR_MSG_UpLoadDictToCloudStatus, indicate cloud personalize data upload success or not; if the last one parameter bOnlyUploadToCloud is not 0, then: every time after call ISSSRUpLoadDict interface, return meaasge ISS_SR_MSG_UpLoadDictToCloudStatus, indicate cloud personalize data upload success or not.

☐ **Parameter specification**

Param Name	Param Explanation
hISSSR [in]	Recognition handle
szList [in]	Grammar message，Json form，Refer to appendix IV，UTF8 coding
bOnlyUploadToCloud [in]	Upload cloud or not

☐ **return value**

1、 ISS_SUCCESS：add a asynchronous compile task success；

2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；

3、 ISS_ERROR_INVALID_HANDLE：invalid recognition handle；

4、 ISS_ERROR_INVALID_PARA：invalid parameter value；

NB：asynchronous interface。Call after handle created, ISSSRUpLoadDict upload dictionary

2.4 Callback interface specification

2.4.1 Recognition callback interface

☐ **interface declaration**

```
typedef void (ISSCALLBACK *Proc_OnMsgProc)(void* pUsrArg,unsigned int uMsg,unsigned int wParam,void *lParam);
```

☐ **interface speciation**

1、 Recognition callback interface；

☐ **Parameter specification**

Param Name	Param Explanation
pUsrArg	User-defined parameter

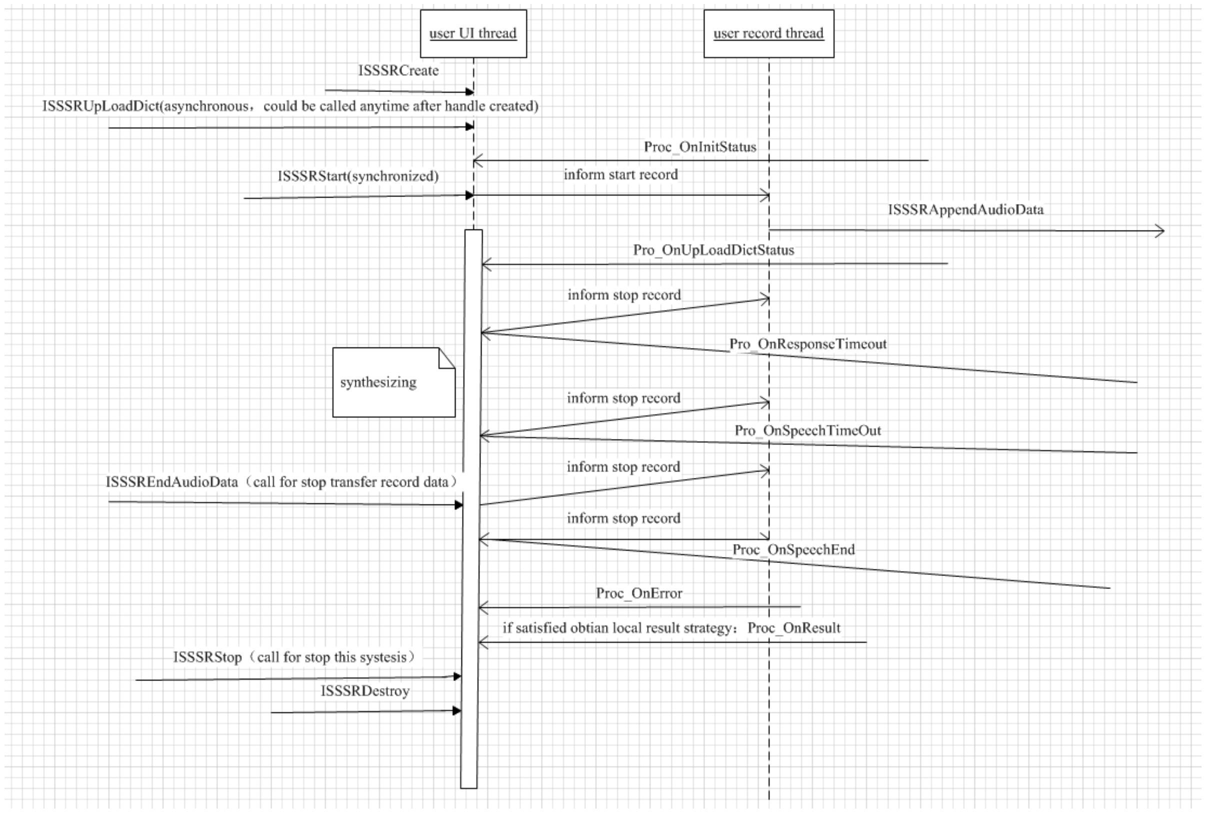
uMsg	Message type introduction

event	introduction
ISS_SR_MSG_InitStatus	wParam: ISS_SUCCESS: initialization success ISS_ERROR_FAIL: initialization fail ISS_ERROR_OUT_OF_MEMORY: run out of memory lParam: NULL
ISS_SR_MSG_UpLoadDictToLocalStatus	wParam: ISS_SUCCESS: local personalized command upload success ； ISS_ERROR_INVALID_JSON_FMT: Error input Json form when compile grammar. ISS_ERROR_INVALID_JSON_INFO: necessary grammar message extracted fail from Json when compile grammar lParam: NULL
ISS_SR_MSG_UpLoadDictToCloudStatus	wParam: ISS_SUCCESS: cloud personalized data upload success ； ISS_ERROR_INVALID_JSON_FMT: Error input Json form when compile grammar. ISS_ERROR_INVALID_JSON_INFO: necessary grammar message extracted fail from Json when compile grammar lParam: NULL
ISS_SR_MSG_VolumeLevel	wParam: max volume， minimum 0， maximum 931 lParam: NULL
ISS_SR_MSG_ResponseTimeout	wParam: NULL lParam: NULL
ISS_SR_MSG_SpeechStart	wParam: NULL lParam: NULL
ISS_SR_MSG_SpeechTimeOut	wParam: NULL lParam: NULL
ISS_SR_MSG_SpeechEnd	wParam: NULL lParam: NULL
ISS_SR_MSG_Error	wParam: error code return by speech recognize engine，Refer to appendix II lParam: const char*type，error message
ISS_SR_MSG_Result	wParam: NULL lParam: const char*type，recognition result

☐ **return value**

1、 Null

2.5 Interface call progress



4. Speech synthesis

3.1 Interface introduction

3.1.1 List of methods

☐ applied interface：

Method Name	Function specification
ISSTTSInitRes	Create speech synthesize resource handle
ISSTTSInitResW	Create speech synthesize resource handle
ISSTTSUnInitRes	Destroy speech synthesize resource handle
ISSTTSCreate	Create speech synthesize handle
ISSTTSDestroy	Destroy speech synthesize handle
ISSTTSSetParam	Set synthesize parameters

ISSTTSStart	Synthesis specious text
ISSTTSGetAudioData	Get audio data
ISSTTSStop	Stop synthesize

3.1.2 Function specification

1、 Speech synthesize support audio string play and synthesize;

3.2 Synthesize parameters specification

1 List table of parameters， if not set， please use default value：

Name	Param	Range of param value
Speaker	ISS_TTS_PARAM _SPEAKER	Refer to Appendix : list of Speaker
Voice speed	ISS_TTS_PARAM _VOICE_SPEED	#define ISS_TTS_SPEED_MIN (-32768) /* minimum voice speed */ #define ISS_TTS_SPEED_NORMAL_DEFAULT (0) /* general voice speed, default value */ #define ISS_TTS_SPEED_MAX (+32767) /* maximum */
tone	ISS_TTS_PARAM _VOICE_PITCH	#define ISS_TTS_PITCH_MIN (-32768) /* minimum tone */ #define ISS_TTS_PITCH_NORMAL_DEFAULT (0) /* general tone, default tone */ #define ISS_TTS_PITCH_MAX (+32767) /* maximum tone */
volume	ISS_TTS_PARAM _VOLUME	#define ISS_TTS_VOLUME_MIN (-32768) /* minimum volume */ #define ISS_TTS_VOLUME_NORMAL (0) /* general volume */ #define ISS_TTS_VOLUME_MAX_DEFAULT (+32767) /* maximum volume , default value*/

3.3 Interface specification

3.3.1 Create speech synthesize resource handle

☐ Function prototype

ISSErrID *ISSAPI* ISSTTSInitRes(HISSTTSRES*phISSTTSRES, const char*szResourceDir,int iResMode);

ISSErrID *ISSAPI* ISSTTSInitResW(HISSTTSRES* phISSTTSRES,const wchar_t* szResourceDir, int iResMode);

☐ **Function specification**

1、 Create speech synthesize resource handle；

☐ **Parameters specification**

Param Name	Param Explanation
phISSTTSRES [out]	TTS resource handle
szResourceDir [in]	Resource directory , in general: ISSTTSInitRes version，under windows Chnises environment is GBK coding, under linux environment is UTF-8 coding；ISSTTSInitResW version， under windows vc environment is UNICODE-16 coding, under linux gcc environment UNICODE-32coding。

iResMode [in]	Loading method:
	iResMode = 0 open file handle;
	iResMode = 1 load into Ram。

☐ **return value**

1、 ISS_SUCCESS：Create speech synthesize resource handle；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_FILE_NOT_FOUND：relevant resource file not found；
4、 ISS_ERROR_INVALID_PARA：invalid parameter value；
5、 ISS_ERROR_OUT_OF_MEMORY：run out of memory。

3.3.2 Destroy speech synthesize resource handle

☐ Function prototype

ISSErrID *ISSAPI* ISSTTSUnInitRes (HISSTTSRES hISSTTSRES);

☐ **Function specification**

1、 Destory speech synthesize resource handle；

☐ **Parameters specification**

Param Name	Param Explanation
hISSTTSRES [in]	TTS resource handle。

☐ **return value**

1、 ISS_SUCCESS：Destroy speech synthesize resource handle；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_INVALID_PARA：invalid parameter value。

3.3.3 Create speech synthesize handle

☐ Function prototype

ISSErrID *ISSAPI* ISSTTSCreate(HISSTTS* phISSTTS, HISSTTSRES hISSTTSRES, Proc_OnTTSDataReady pcbOnTTSDataReady, void* pUsrArg);

☐ **Function specification**

1、 Create speech synthesize handle；

☐ **Parameters specification**

Param Name	Param Explanation
phISSTTS [out]	TTS handle
hISSTTSRES[in]	TTS resource handle
pcbOnTTSDataReady[in]	User OnTTSDataReady callback function pointer，refer to 3.4
pUsrArg[in]	User-define paratmeters，apply user by Callback function

☐ **return value**

1、 ISS_SUCCESS：Create speech synthesize handle；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_FILE_NOT_FOUND：relevant resource file not found；
4、 ISS_ERROR_INVALID_PARA：invalid parameter value；
5、 ISS_ERROR_OUT_OF_MEMORY：run out of memory。

NB：Can use same TTS resource create multi TTS object.

3.3.4 Destroy speech synthesize handle

☐ Function prototype

ISSErrID *ISSAPI* ISSTTSDestroy(HISSTTS hISSTTS);

☐ **Function specification**

1、 Destroy speech synthesize handle by this interface；

☐ **Parameters specification**

Name	Range of param value
hISSTTS [in]	TTS handle

☐ **Return value**

1、 ISS_SUCCESS：Destroy speech synthesize handle and release resource success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_INVALID_HANDLE：invalid speech synthesize handle。

3.3.5 Parameters configure

☐ Function prototype

ISSErrID *ISSAPI* ISSTTSSetParam(HISSTTS hISSTTS,int iParamID,int iParamValue);

☐ **Function specification**

1、 Set speech synthesize parameters，include：synthesize speaker、speech speed and tone；

☐ **Parameters specification**

Param Name	Param Explanation
hISSTTS [in]	TTS handle。
iParamID [in]	TTS parameters index。
iParamValue [in]	Relevant parameter value 。

☐ **return value**

1、 ISS_SUCCESS：set parameters success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_INVALID_HANDLE：invalid speech synthesize handle；
4、 ISS_ERROR_INVALID_PARA：invalid parameter type；
5、 ISS_ERROR_INVALID_PARA_VALUE：invalid parameter value。

3.3.6 Synthesis of specified text

☐ Function prototype

ISSErrID *ISSAPI* ISSTTSStart(HISSTTS hISSTTS,const void* pText,int iSize,ISSTTSCodePage iTTsCodePage);

☐ **Function specification**

1、 Synthesis specified text, if engine are doing one text，but call SSTTSStart synthesis another text，the first will be abort。

☐ **Parameters specification**

Param Name	Param Explanation
hISSTTS [in]	TTS handle
pText [in]	Synthetic text Buffer
iSize [in]	Synthetic text Buffer size
iTTsCodePage [in]	Synthetic text code

☐**return value**

1、 ISS_SUCCESS：start synthesis success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_INVALID_HANDLE：invalid speech synthesize handle；
4、 ISS_ERROR_INVALID_PARA：invalid parameter value。

3.3.7 Obtain synthesis audio

☐ Function prototype

ISSErrID *ISSAPI* ISSTTSGetAudioData(HISSTTS hISSTTS,void *pOutBuffer,unsigned int iBufferFrames,unsigned int *piBufferFramesGot,unsigned int *piProgBegin,unsigned int *piProgLen);

☐ **Function specification**

1、 Obtain audio data，support 16000sample rate、S16-LE、Mono audio，size of a sample point is 2Bytes;

☐ **Parameters specification**

Param Name	Param Explanation
hISSTTS [in]	TTS handle
pOutBuffer [in]	Audio output Buff address
iBufferFrames [in]	Expect obtain sample points, non-byte number
piBufferFramesGot [out]	The actual output sample rate
piProgBegin[out]	Current progress offset , reserved , not implemented
piProgLen[out]	Current progress length , reserved, not implemented

☐ **Return value**

1、 ISS_SUCCESS：Obtain synthesis speech success;

2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II;

3、 ISS_ERROR_INVALID_HANDLE：invalid speech synthesize handle;

4、 ISS_ERROR_INVALID_PARA：invalid parameter value;

5、 ISS_ERROR_TTS_STOPPED ： error called ， synthes is over ， but call ISSTTSGetAudioData obtain record。

6、 ISS_ERROR_TTS_COMPLETED：obtain all synthesis speech，synthesis over。

NB：Synchronous blocking function

3.3.8 Cancel synthesis

☐ Function prototype

ISSErrID *ISSAPI* ISSTTSStop(HISSTTS hISSTTS);

☐ **Function specification**

1、 Stop synthesis。

☐ **Parameters specification**

Param Name	Param Explanation
hISSTTS [in]	TTS handle

☐ **Return value**

1、 ISS_SUCCESS：Obtain synthesis speech success;

2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II;

3、 ISS_ERROR_INVALID_HANDLE：invalid speech synthesize handle;

4、 ISS_ERROR_TTS_STOPPED：error called，synthes is over。

3.4 Callback interface declaration

3.4.1 Synthesis resource initializes success

☐ **interface declaration**

typedef void (ISSCALLBACK* Proc_OnTTSDataReady) (void* pUsrArg);

☐ **interface introduction**

1、 Callback inform after synthesis，call ISSTTSGetAudioData obtain audio data；

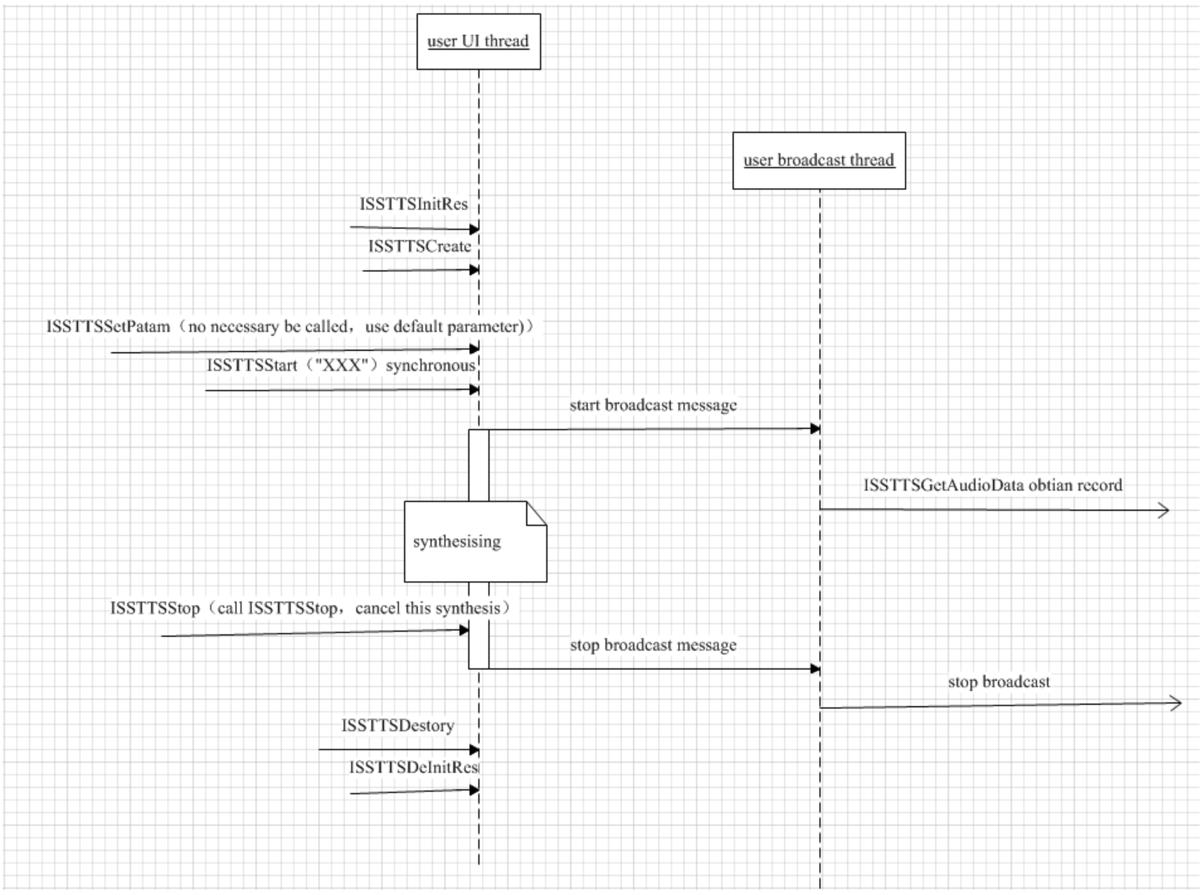
☐ **Parameters specification**

Param Name	Param Explanation
pUsrArg [in]	User-defined parameter

☐ **Return value**

1、 Null；

3.5 Interface call progress


user UI thread
user broadcast thread
ISSTTSInitRes
ISSTTSCreate
ISSTTSSetPatam（no necessary be called，use default parameter)）
ISSTTSStart（"XXX"）synchronous
start broadcast message
ISSTTSGetAudioData obtian record
synthesising
ISSTTSStop（call ISSTTSStop，cancel this synthesis）
stop broadcast message
stop broadcast
ISSTTSDestory
ISSTTSDeInitRes

5. Speech wake-up

4.1 Interface declaration

4.1.1 list of method

☐ list of applied interface：

Param Name	Param Explanation
ISSVWCreate	Create speech wake-up handle
ISSVWCreateW	Create speech wake-up handle
ISSVWDestroy	Destroy speech wake-up handle
ISSVWSetThreshold	Set wake-up threshold
ISSVWStart	Start speech wake-up, if speech wake-up is started ,always return ISS_SUCCESS
ISSVWAppendAudioData	Input audio record data, support 16K sample rate、S16-LE、Mono PCM record
ISSVWStop	Stop speech wake-up, if wake-up stopped, always return ISS_SUCCESS

4.1.2 Function declaration

1、 Speech wake-up，local engine model；

4.2 Wake-up Parameter

Null。

4.3 Interface declaration

4.3.1 Create speech wake-up handle

☐ Function prototype

ISSErrID *ISSAPI* ISSVWCreate (HISSVW *phISSVW, const wchar_t * szResourceDir, Proc_ISSVWOnWakeup pfnOnWakeup, void* pUsrArg);

ISSErrID *ISSAPI* ISSVWCreateW (HISSVW *phISSVW, const wchar_t * szResourceDir, Proc_ISSVWOnWakeup pfnOnWakeup, void* pUsrArg);

☐ **Function specification**

1、 Create speech wake-up handle；

☐ **Parameters specification**

Param Name	Param Explanation
phISSVW [out]	Wake-up handle
szResourceDir[in]	Resource directory, in general：ISSSRCreate version，under windows Chinese enviroment is GBKcoding，under linux environment is UTF-8 coding；ISSSRCreateW version， under windows vc environment is UNICODE-16 coding， under linux gcc environment is UNICODE-32coding。
Proc_ISSVWOnWakeup [in]	Recognition callback interface， reference in Identify callback interface declaration， cannot pass NULL；
pUsrArg [in]	User-defined parameter

☐ **Return value**

1、 ISS_SUCCESS：Create speech wake-up handle success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_FILE_NOT_FOUND：error found relevant file ；
4、 ISS_ERROR_INVALID_PARA：invalid parameter value；
5、 ISS_ERROR_OUT_OF_MEMORY：run out of memory；
6、 ISS_ERROR_INVALID_DATA：error by resource data。

4.3.2 Destroy speech wake-up handle

☐ Function prototype

ISSErrID *ISSAPI* ISSVWDestroy(HISSVW hISSVW)

☐ **Function specification**

1、 Destroy speech wake-up handle；

☐ **Parameters specification**

Param Name	Param Explanation
hISSVW [in]	Wake-up hanlde

☐ **Return value**

1、 ISS_SUCCESS：Destroy speech wake-up handle and release resource success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_INVALID_HANDLE：invalid speech wake-up handle。

4.3.3 Configure speech wake-up threshold

☐ Function prototype

ISSErrID *ISSAPI* ISSVWSetThreshold(HISSVW hISSVW,int nThreshold);

☐ **Function specification**

1、 Configure speech wake-up threshold；

☐ **Parameters specification**

Param Name	Param Explanation
hISSVW [in]	Wake-up handle
nThreshold [in]	threshold

☐ **Return value**

1、 ISS_SUCCESS：Configure threshold success；

2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；

3、 ISS_ERROR_INVALID_HANDLE：invalid speech wake-up handle；

4.3.4 Activate speech wake-up

☐ Function prototype

ISSErrID *ISSAPI* ISSVWStart(HISSVW hISSVW)

☐ **Function specification**

1、 Activate speech wake-up；

2、 Interfaces are synchronous；

3、 If wake-up active, always return ISS_SUCCESS。

☐ **Parameters specification**

Param Name	Param Explanation
hISSVW [in]	Wake-up handle

☐ **Return value**

1、 ISS_SUCCESS：Activate speech wake-up success；

2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；

3、 ISS_ERROR_INVALID_HANDLE：invalid speech wake-up handle。

4、 ISS_ERROR_INVALID_CALL：error called, wake-up was start。

4.3.5 Input audio record data

☐ Function prototype

ISSErrID *ISSAPI* ISSVWAppendAudioData(HISSVW hISSVW,short* pSamples,unsigned int nNumberOfToWrite,unsigned int* pNumberOfWritten);

☐ **Function specification**

1、 Input audio record data；

2、 Support 16K sample rate、S16-LE、Mono PCM record；

3、 Only can be called in one thread.

☐ **Parameters specification**

Param Name	Param Explanation
hISSVW [in]	Wake-up handle
pSamples [in]	Output audio buffer address；
nNumberOfToWrite [in]	The number of sample point need to read-in, but not bytes
pNumberOfWritten [out]	The actual read-in number of sample points

☐ **Return**

1、 ISS_SUCCESS：read-in record success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_INVALID_HANDLE：invalid speech wake-up handle。；
4、 ISS_ERROR_INVALID_PARA：invalid parameters value ；
5、 ISS_ERROR_INVALID_CALL：error call (call ISSVWStart before speech wake-up activate)。

4.3.6 Stop speech wake-up

☐ Function prototype

ISSErrID *ISSAPI* ISSVWStop(HISSVW hISSVW);

☐ **Function specification**

1、 Stop speech wake-up；

☐ **Parameters specification**

Param Name	Param Explanation
hISSVW [in]	Wake-up handle

☐ **Return value**

1、 ISS_SUCCESS：stop speech wake-up success；
2、 ISSErrID：error code return by speech recognize engine，Refer to appendix II；
3、 ISS_ERROR_INVALID_HANDLE：invalid parameters value。
4、 ISS_ERROR_INVALID_CALL: error call (call ISSVWStart before speech wake-up activate);

4.4 Callback interface specification

4.4.1 Wake-up success

☐ **Interface declaration**

typedef void (ISSCALLBACK *Proc_ISSVWOnWakeup) (void* pUsrArg);

☐ **Interface specification**

2、 Wake-up callback interface；

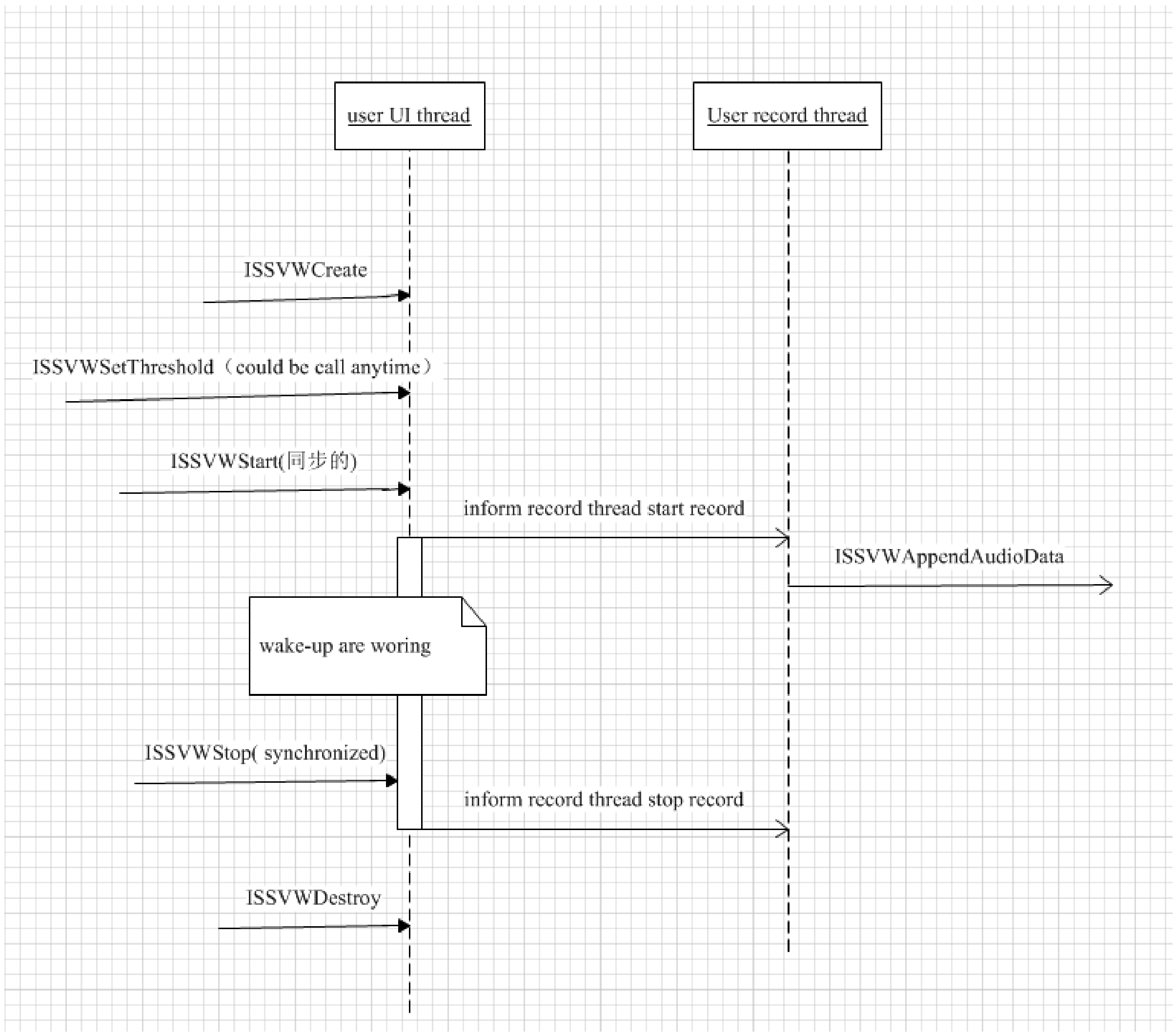
☐ **Parameters specification**

Param Name	Param Explanation
pUsrArg [in]	User-define parameters

☐ **Return value**

2、 Null；

4.5 Interface call progress



6. Suite Authorization

5.1 Interface introduction

5.1.1 List of functions

☐ supported functions：

Method Name	Function Introduction
ISSSetMachineCode	Input methane code
ISSGetActiveKey	Obtain active key
ISSActivate	Active suite by machine code and active key

5.1.2 Function introduction

1、 Suite authorization manage interfce，there are two ways to active suite：software encryption and hardware encorption。

2、 software encryption use ISSGetActiveKey + ISSActivate interface，obtain active key，only need to be called one time while join internet ； hardware encryption use ISSSetMachineCode interface。

5.2 Authorization Parameters introduction

Null。

5.3 Interface Specification

5.3.1 Set machine code

☐ Function prototype

ISSErrID ISSSetMachineCode(const char* szMachineCode);

☐ **Function specification**

1、 Input machine code，hardware authorization interface；

☐ **Parameter specification**

Param Name	Param Explanation
szMachineCode [in]	Machine code

☐ **Return value**

1、ISS_SUCCESS：input machine code success；

2、ISSErrID：error of speech suite authorization active，Refer to appendix II；
3、ISS_ERROR_INVALID_PARA：invalid parameter value；
4、ISS_ERROR_FAIL:input machine code fail;

NB：before call ISSGetActiveKey、before call ISSActivate、and before call ISSSRCreate, need to call this interface at the least one time。Every client need to input different machine code，machine code no more longer than 1024 characters。Machine code must be unique，use for flag personalized resource 。

5.3.2 Obtain active key

☐ Function prototype

ISSErrID ISSGetActiveKey(const char* szResourceDir);

☐ **Function specification**

1、 Obtain active key；

☐ **Parameter specification**

Param Name	Param Explanation
szResourceDir [in]	Resource directory ,in general： Under windows Chinese environment is GBK coding， Under linux environment is UTF-8coding；

☐ **Return value**

1、ISS_SUCCESS：obtain active code success；
2、ISSErrID：error of speech suite authorization active，Refer to appendix II；
3、ISS_ERROR_FAIL：obtain active code fail；
4、ISS_ERROR_MACHINE_CODE_NOT_SET: need to set machine code;
5、ISS_ERROR_INVALID_MACHINE_CODE：the machine not authorized；
6、ISS_ERROR_ACTIVATE_TO_MANY_TIMES：over authorized times；
7、ISS_ERROR_NET_XXXX：network error

NB：Obtain active key，need to be called one times while join internet，before call ISSGetActiveKey, must call ISSSetMachineCode configure machine code。

5.3.3 Activate

☐ Function prototype

ISSErrID ISSActivate(const char* szResourceDir);

☐ **Function specification**

1、 input active code directory , active；

☐ **Parameter specification**

Param Name	Param Explanation
szResourceDir [in]	Resource directory ,in general： Under windows Chinese environment is GBK coding， Under linux environment is UTF-8coding；

☐ **Return value**

1、ISS_SUCCESS：Authorization active success；

2、ISSErrID：error of speech suite authorization active，Refer to appendix II；

3、ISS_ERROR_INVALID_PARA：invalid parameter value；

4、ISS_ERROR_INVALID_ACTIVE_KEY：invalid active key；

5、ISS_ERROR_MACHINE_CODE_NOT_SET：need to set machine code;

NB：before call ISSSRCreate、before call ISSTTSInitRes, need to call this interface at the least one time。Before call ISSActivate, must call ISSSetMachineCode configure machine code。

7. Interface used example

7.1 Scene 1

Play warming tone before Speech recognition record，remind user start input speech，Can be used in combined recognition and synthesis：

```
#define PCM_16K_BUFFER_SIZE (320)
//return value
int res;

//wake-up handle
HISSVW *phISSVW = NULL;

//input wake-up data
Char *data_buff = new char[PCM_16K_BUFFER_SIZE];

//wake-up callback
static void OnWakeUp(void *);

//create wake-up handle
res = ISSVWCreate(phISSVW,"\storage card\", OnWakeUp,NULL);

//configure wake-up threshold
res=  ISSVWSetThreshold(phISSVW,0);
```

```
// start wake-up
Res = ISSVWStart(phISSVW);

//input record model
Res = IvwAppendAudioData(phISSVW, data_buff,PCM_16K_BUFFER_SIZE/2);
... ... ...
//speech wake-up success
Static void OnWakeUp(void *)
{
    //create and initialize recognize handle
    // create and initialize synthesis handle
    ... ... ...
    While（1）
    {
       If(ISS_ERROR_TTS_COMPLETED != ISSTTSGetAudioData(...))
       {
          //play obtain audio (warming tone)
       }
       Else
       {
          //after warming tone，recognize and start record
          Res = ISSSRStart(...);
          Break;
        }
       Sleep(100);
    }
}
```

7.2 Scene 2

In interact ，start choose scene，for example in case of telephone，call ZhanSan，there more than one contact，will support one time interact for user to choose target contact，at the this time need to use scene recognition：

```
//TTS state enum
typedef enum  _tagTTSState
{
    TTS_State_Init,                           //original state
    TTS_State_Record,                         //need to record after report
    TTS_State_Record_Contacts,                // need to record after report
                                              //contact scene

    TTS_State_Record_SMS,                     // need to record after report
                                              //message scene

    TTS_State_Select,                         //selected scene
    TTS_State_Record_Confirm,                 //determine scene
}TTSState;

//return value
Int Res;

//current TTS state
TTSState m_ttsstate;

//multi contact, remind user choose target one
m_ttsstate = TTS_State_Select;
Res =  ISSTTSStart("choose one to call…");
...

//after report, start choose scene recognition
void OnTTSCompleted()
{
   switch(m_ttsstate)
   {
     ...
     Case TTS_State_Select:
     {
        Res = ISSSRStart(hISSSRObj,"select",2,NULL);
     }
     ...
   }
}
```

8.Technical support

Please contact us if encounter any question.

Tell us details when contacting：

* system configuration （CPU, RAM, OS and version information etc.）

* details of problem （process of problem recurring , how the problem display etc.）

*details of operation steps during developing

Telephone Support

From Monday to Friday，Beijing time 9：00～17：00

Call：18805691018

Email Support

Send questions to ：autoflySupport@iflytek.com。

Appendix I： Personalize speaker list

1、 Speaker of English & Chinese: support mix reading.
2、 English speaker only broadcasts English. N/A for Chinese.
3、 Chinese speaker only broadcasts Chinese. 遇到英文会以单个字母的方式进行朗读。

Name of speaker	Speaker type	Language	Parameters	Mark
Jiajia	Female Child	Chinese&English(Chinese Mandarin)	ivTTS_ROLE_JIAJIA	
Tianchang	Female Youth	Chinese&English(Chinese Mandarin)	ivTTS_ROLE_TIANCHANG	
Wenjing	Female Youth	Chinese&English(Chinese Mandarin)	ivTTS_ROLE_WENJING	
Xiaoyan	Female Youth	Chinese&English(Chinese Mandarin)	ivTTS_ROLE_XIAOYAN	
YanPing	Female Youth	Chinese&English(Chinese Mandarin)	ivTTS_ROLE_YANPING	
XiaoFeng	Male Youth	Chinese&English(Chinese Mandarin)	ivTTS_ROLE_XIAOFENG	
YuFeng	Male Youth	Chinese&English(Chinese Mandarin)	ivTTS_ROLE_YUFENG	
Sherri	Female Youth	US English	ivTTS_ROLE_SHERRI	
Xiaojin	Female Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_XIAOJIN	
Nannan	Male Child	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_NANNAN	
Jinger	Female Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_JINGER	
Yuer	Female Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_YUER	
Xiaoqian	Female Youth	Chinese (Dongbei accent)	ivTTS_ROLE_XIAOQIAN	
Laoma	Male Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_LAOMA	
Bush	Male Youth	US English	ivTTS_ROLE_BUSH	
Xiaorong	Female Youth	Chinese （Sichuan accent）	ivTTS_ROLE_XIAORONG	
Xiaomei	Female Youth	Chinese (Cantonese)	ivTTS_ROLE_XIAOMEI	

Anni	Female Youth	Chinese&English(Chinese Mandarin)	ivTTS_ROLE_ANNI	
John	Male Youth	US English	ivTTS_ROLE_JOHN	
Anita	Female Youth	British English	ivTTS_ROLE_ANITA	
Terry	Female Youth	US English	ivTTS_ROLE_TERRY	
Catherine	Female Youth	US English	ivTTS_ROLE_CATHERINE	
Terry	Female Youth	US English Word	ivTTS_ROLE_TERRYW	
Xiaolin	Female Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_XIAOLIN	
Xiaomeng	Female Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_XIAOMENG	
Xiaoqiang	Male Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_XIAOQIANG	
XiaoKun	Male Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_XIAOKUN	
Jiu Xu	Male Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_JIUXU	
Duo Xu	Male Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_DUOXU	
Xiaoping	Female Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_XIAOPING	
Donald Duck	Male Youth	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_DONALDDUCK	
Baby Xu	童年男声	Chinese&English (Chinese Mandarin)	ivTTS_ROLE_BABYXU	
Dalong	Male Youth	Chinese（Cantonese）	ivTTS_ROLE_DALONG	
Tom	Male Youth	US English	ivTTS_ROLE_TOM	

Appendix II： Error code list

1、Error code list：

Error code	value	sense
ISS_SUCCESS	0	Success state
ISS_ERROR_FAIL	-1	failure

ISS_ERROR_EXCEPTION	-2	exception
ISS_ERROR_INVALID_CALL	10000	Illegal call
ISS_ERROR_INVALID_JSON_FMT	10001	Invalid JSON form
ISS_ERROR_INVALID_JSON_INFO	10002	Invalid JSON content
ISS_ERROR_TTS_STOPPED	10003	Error call，synthesis over
ISS_ERROR_TTS_COMPLETED	10004	Obtain all synthesis audio，synthesis over
ISS_ERROR_CREATE_THREAD_FAIL	10005	Create thread failure
ISS_ERROR_NO_SPEECH	10006	No speech input
ISS_ERROR_GENERAL	10100	
ISS_ERROR_OUT_OF_MEMORY	10101	Run out of memory
ISS_ERROR_FILE_NOT_FOUND	10102	File no found
ISS_ERROR_INVALID_PARA	10106	Invalid parameter type
ISS_ERROR_INVALID_PARA_VALUE	10107	Invalid parameter value
ISS_ERROR_INVALID_HANDLE	10108	Invalid handle
ISS_ERROR_INVALID_DATA	10109	invalid

Appendix Ⅲ：Recognition result instruction

1、The result is a format of XML. Samples are:

Sample 1 ： Open ANGRY BIRD

```
        <rawtext>open ANGRY BIRD/rawtext>
        <version>1.0</version>
        <result>
                <focus> app </focus>
                <action>
                        <operation> launch</operation>
                </action>
                <object>
                        <name>ANGRY BIRD</name>
                </object>
</result>
```

Sample 2：Call Jack

```
       <rawtext>call Jack</rawtext>
        <version>1.0</version>
        <result>
                <focus>telephone</focus>
                <action>
                        <operation> call</operation>
                </action>
                <object>
                        <name>Jack</name>
                </object>
        </result>
```

In Sample 2，from XML we can see： business is Phone; operation is Call; contact is Jack.

Appendix IV： Personalized data upload format

1、Personalized data form：

```
{
    "grm": [
        {
            "dictname": "contact",
            "dictcontant": [
                { "name": "iFLYTEK", "id": 0 },
                { "name": "Xuebai", "id": 1 },
                { "name": "Junfeng liu",      "id": 2 },
                { "name": "Yawei Bai", "id": 3 }
            ]
        },
        {
            "dictname": "singers",
            "dictcontant": [
                { "name": "Fenghuangchuanqi", "id": 0 },
                { "name": "Yina na", "id": 1 },
                { "name": "Kun Yang", "id": 2 },
                { "name": "S.H.E", "id": 3 }
            ]
        },
        {
            "dictname": "songs",
            "dictcontant": [
                { "name": "Gong xi fa cai", "id": 0 },
                { "name": "Pian ai", "id": 1 },
                { "name": "Keng qiang mei gui", "id": 2 },
                { "name": "Gu zhen nan mian", "id": 3 }
            ]
        },
        {
            "dictname": "apps",
            "dictcontant": [
                { "name": "Xun fei yu dian", "id": 0 },
                { "name": "angry bird", "id": 1 },
                { "name": "Brower", "id": 2 }
            ]
        }
    ]
}
```

2、Data form is Json，UTF8 coding。**Dictname** label indicate upload personalized dictionary type，so far support（contact)，(singers)，(songs）and（apps） four types。Dictcontant indicate upload data，name sub label indicate data name，id indicate ID number of data，example，after contact data upload successful，then contact uploaded can be recognized，call xuebai、call Yawei bai and so on。

3、Personalized data upload interface ISSSRUpLoadDict(HISSSR hISSSR, const char *szList，int bOnlyUploadToCloud)，parameter bOnlyUploadToCloud indicate upload successful or not；during the processing of upload personalized data，if due to internet situation cause upload failure ，interface will not return ISS_SR_MSG_UpLoadDictToCloudStatus message，developer could record logs，after internet linked then set bOnlyUploadToCloud to be TRUE，personalized data upload to internet server，to avoid problem when upload failure lead to recognition model switch to internet recognition and personalized data cannot reorganization.

Appendix V： Personalized command word instructions

1、under wince.arm\bin\SRRes\could find isssr.cfg，
example：

```
{
    "Appid" : "526664b0",
    "Cmds" : [
        { "name": "the song name",      "id": 1 },
        { "name": "what is the song name",       "id": 1 },
        { "name": " How far is it to the destination ",   "id": 2 }
    ]
}
```

，Cmds label indicate uploaded personalized command word。Name indicate name of command word ，id indicate relevant id number，same id indicate that the command is the same，for example，what is the song name and what is the name of the playing music，their command id is 1，indicate the semantic are same。

Semantic parser form ：

```
<?xml version="1.0" encoding="utf-8"?>
<nlp>
    <version>1.0.0.2600</version>
    <rawtext> what is the name of the playing music </rawtext>
    <parsedtext> what is the name of the playing music </parsedtext>
    <result>
        <focus>cmd</focus>
```

```
            <object>
                <name> what is the name of the playing music </name>
                <id>1</id>
            </object>
        </result>
</nlp>

<?xml version="1.0" encoding="utf-8"?>
<nlp>
        <version>1.0.0.2600</version>
        <rawtext>what is the song name</rawtext>
        <parsedtext> what is the song name </parsedtext>
        <result>
            <focus>cmd</focus>
            <object>
                <name> what is the song name </name>
                <id>1</id>
            </object>
        </result>
</nlp>
```

These command word expert <id>*</id>，other label are same，parser base on id，same id indicate same meaning， execute same application command。