ViGo® Developer Guide

Mobile application development with ViGo

Title: ViGo Developer Guide

Part number: VV/VIGO/DOC/189/B

VoiceVault Inc.

400 Continental Blvd

6th Floor

El Segundo

CA 90245

USA

(310) 426 2792

info@voicevault.com

Table of Contents

Introduction

ViGo has been specifically designed to allow rapid development of voice authentication solutions in your mobile applications. Supporting the most popular mobile platforms, iOS and Android, your development team can now build new applications that leverage our best-in-class voice biometric systems, or quickly extend your existing apps to incorporate the out-of-the-box convenience and security that ViGo delivers for your customers.

Overview

When you purchase ViGo, you gain access to our high-availability, tried and tested voice biometric infrastructure directly from your mobile application and across both iOS and Android platforms.

All the code to enable connectivity to the ViGo servers, to manage voice registration (enrollment) and voice logon (verification) for users, and for device specific functionality such as recording and manipulating audio samples, is provided by a simple ViGo library. Implementing voice biometric security then becomes as easy as adding this library and incorporating the simple source code snippets provided in to your application. Developing ViGo mobile apps can now be achieved in a matter of hours, rather than the weeks it took to develop using similar technologies in the past.

Developing for ViGo

Prerequisites

Developing a mobile app to use ViGo voice biometrics just requires your ViGo application credentials, which were provided when you signed up. These are inserted in to the relevant places in the source code provided, and the ViGo library added to your application project. You can develop your app using the standard tools available for your platform, such as Xcode for iOS and Eclipse for Android.

The credentials you receive in the welcome email when you register include the following parameters:

VIGO_SERVER_URL	The URL string for the ViGo server that will support your application
VIGO_APP_ID	A unique ViGo identifier for your application
VIGO_CREDENTIAL_ID	The credential string that secures all connections to the ViGo server.
VIGO_CREDENTIAL_PWD	A password corresponding to the credentials for securing server access
CONFIGURATION_ID	ViGo configuration identifiers for the phrases used by users of your app. Note that a number of these are provided to you for the different phrases and modes supported by ViGo.

It is important to note that the server URL, application identifier and credentials are specific to your mobile application only, and should not be shared between other applications. However, they can safely be used across platforms in the same application, so are shared by Android and iOS versions of the same app for instance. We advise that these access credentials are protected in the same way as you would with any sensitive information and not share them outside the direct development team that need to include them in the app itself.

Once you have received your welcome email and ViGo application credentials, you can move on to planning how you incorporate ViGo in to your app.

ViGo Biometric Functionality

User registration

As with any security mechanism that you build in your app, whether it is based on passwords or biometrics, it is first necessary to voice register the user of the app with the system.

How this is implemented in your system is highly dependent on the workflow of your application and the functionality that you are trying to achieve. With ViGo, the voice registration process requires the recording and submission of the user speaking the desired short phrase a few times. This is typically four times for digit mode and three times for phrases.

The choice of registration phrase is up to you and is determined by the specific ViGo configuration identifier, and a set of these is provided with your credentials when you register for ViGo on the VoiceVault website.

Each user of your app who voice registers successfully will be identified with a unique claimant identifier that you must persist to use for future secure interaction by that person with your app. The voice registration of a user only ever needs to be done once with ViGo.

All the functionality required to voice register users with the ViGo system, including recording and submitting the speech itself, is included in the library provided.

User voice logon

Once a user of your app has registered their voice with ViGo, then you can use their claimant identifier to allow them to voice logon and access your application securely and conveniently for all future interactions.

As well as for logging in to your app, you can use ViGo voice biometrics to, for example, authorizes transactions or to supplement or replace passwords anywhere in your app.

A voice logon will typically require that the user speak the short phrase or digit string that they registered with just a once, although further requests may be made by ViGo if further speech is required to ensure the high level of accuracy that ViGo provides.

All functionality needed to provide voice logon and access control for users of your application is included in the ViGo library.

Voice model adaptation

In order to ensure that your users always have the best experience while using your application with ViGo, it is important that their voice model is kept fresh and up to date.

It is likely that a mobile user will access your applications from many environments with different sound characteristics and varying amounts of noise. Additionally your user's voice will vary over time, and even throughout the day and as their mood changes.

ViGo has been designed specifically to ensure that they are still able to use your application consistently and without compromising security despite these external and personal characteristic variations. This is achieved by continually tuning the user's voice model as they interact with the ViGo servers, a process known as *voice model adaptation*.

In order to provide the most effective adaptation performance, ViGo provides two separate mechanisms; automatic and supervised both of which are mandatory in a ViGo-enabled app.

Automatic adaptation

This method of adaptation of your users' voice models is carried out automatically by the ViGo servers every time they interact with the system. This means that there is no need to develop any code or implement any workflow in your application to gain the advantages of continual adaptation for every user. Automatic server-side adaptation is effective at ensuring a user's voice model is kept up to date over time.

Supervised adaptation

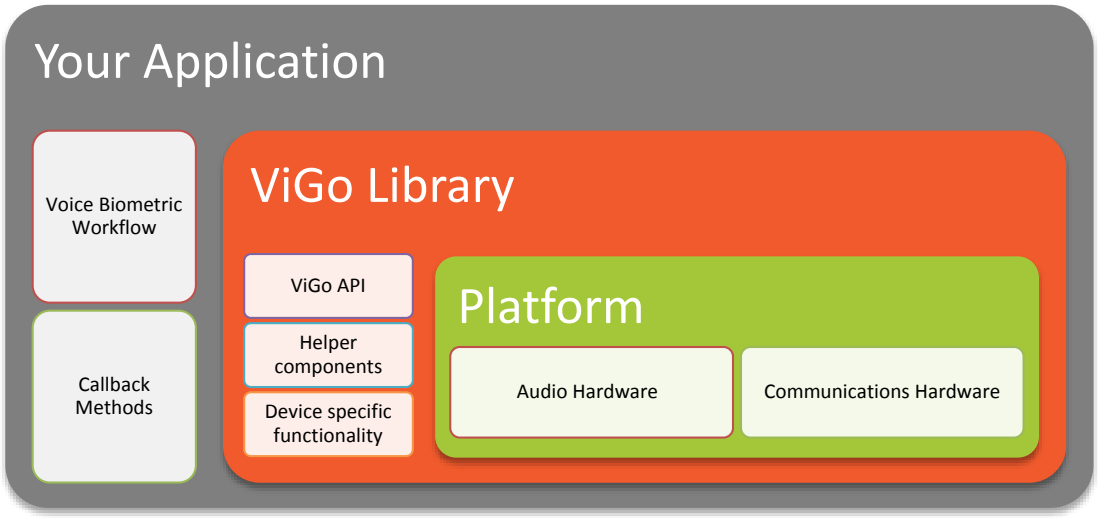
This method of adaptation must be actioned by your application, through a specific method in the ViGo API. It is used to adapt a voice model in the specific instances when the user may not be able to satisfy the high security level enforced by the ViGo system, due to temporary environmental issues such as noise or when there is a more permanent alteration in the user's environment or voice characteristics.

While supervised adaptation is very effective, it requires that a mechanism be made available in the app to allow the user to prove that they are who they say they are. This could be through the entry of a simple PIN or swipe pattern that the user was required by the app to provide when voice registering, as demonstrated in the VoiceVault ViGo demo app, or it could be something specific to your own application such as entry of personally held information. Either way, it is up to the developer to design this mandatory functionality into the app.

After successfully confirming the user identity by providing this additional factor, the application code must then call the supervised adaptation method in the ViGo API so that the voice model is adapted based on the last user interaction. This is the speech that did not quite reach the highest security levels required for the user to logon, but which still far exceeds the level that any potential imposter would be able to achieve.

Library and Sample Code

When you sign up to ViGo you will be provided with the library and sample code snippets for your platform(s). The library is added to your application project through your IDE (Xcode or Eclipse) in the usual way, and the sample code snippets included in to your workflow in the relevant place dependent on your desired functionality.



ViGo Library

The library provides the API for ViGo specifically designed for your platform, along with other helper methods that make supporting voice biometric functionality in your application quick and straightforward.

Once you have planned how the voice registration and logon will fit in to your application workflow, you insert the provided code snippets that call the relevant ViGo API methods and corresponding callbacks in to the app. By leveraging the provided library and code, it is possible to develop a simple ViGo enabled application in a matter of hours.

Sample App

Full source code for a rudimentary app that demonstrates voice registration and login is available from the VoiceVault ViGo website.

To run the app, you will first need to update the library search paths for your specific environment and insert the access credentials and desired digit or phrase configuration identifier as described above.

The app takes you through a very simple voice registration / voice login process as follows:

1. Click the Registration button (the Login button is disabled until voice registration has taken place)
2. You will then see the voice prompt screen that includes the phrase or digit string associated with the configuration identifier you chose to use, and the green Record button. You will need a microphone to be enabled at this point.
3. Click the green Record button and say the phrase or digit string on the screen. The Record button will be greyed out until it is time for you to say the phrase / digit string again.
4. Keep repeating the above step until the system has gathered sufficient voice samples to register your voice. This will typically be three for phrases and four for digits.
5. When voice registration is complete, you will be returned to the main screen and the Login button will be enabled. You can now press this button and, using an analogous process to that used for voice registration, verify your identity. This will typically require one voice sample.

You are free to use / modify this code as you see fit or to use it as the basis for adding Vigo functionality to your app. Note however that the mandatory adaptation processes are not included in the sample app and will need to be added by you.

Platform support

As part of the ViGo library, VoiceVault have developed and wrapped the platform specific hardware functionality so that developers can focus on building their application. This includes the audio hardware to record the user speech in the correct format for use by ViGo, as well as the communications interface to securely access the ViGo servers.

Using ViGo

A ViGo enabled application can be quickly built for iOS using Xcode and the supplied library and code snippets. In this way it is possible to build class-leading voice biometric support in any new app or enhance your existing apps quickly and easily.

Regardless of whether it is a new or existing application, it is important to plan how to build ViGo in to the workflow to ensure it is convenient and effective for your users.

ViGo workflow

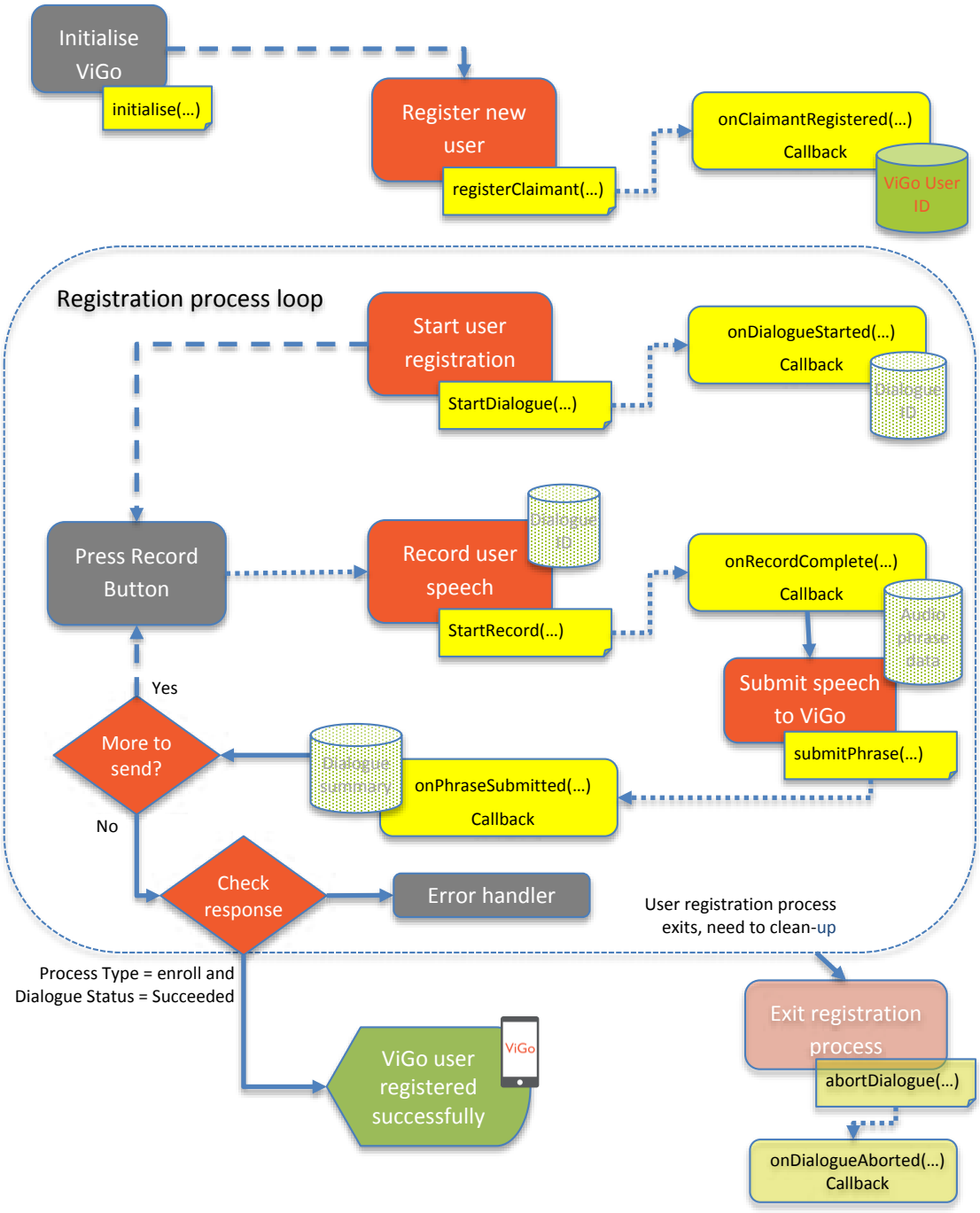
Typically you will need to consider how best to integrate the two separate ViGo operations, namely user voice registration and user voice logon. Depending on the nature of your application, this may be in very different (and multiple) areas of your application.

Once the integration in to your application has been determined, it is only necessary to insert the relevant calls to the ViGo API in your code. In order to ensure that the impact to your application is kept to the minimum, the ViGo library code follows best practices for each platform that it supports.

- Code running on the main (UI) thread is kept to a minimum
- Application lifecycle management requirements are considered
- Responses from the ViGo system are returned through callbacks
- Separate threads are used to support longer running functions
- Services are implemented where appropriate for all communication functionality

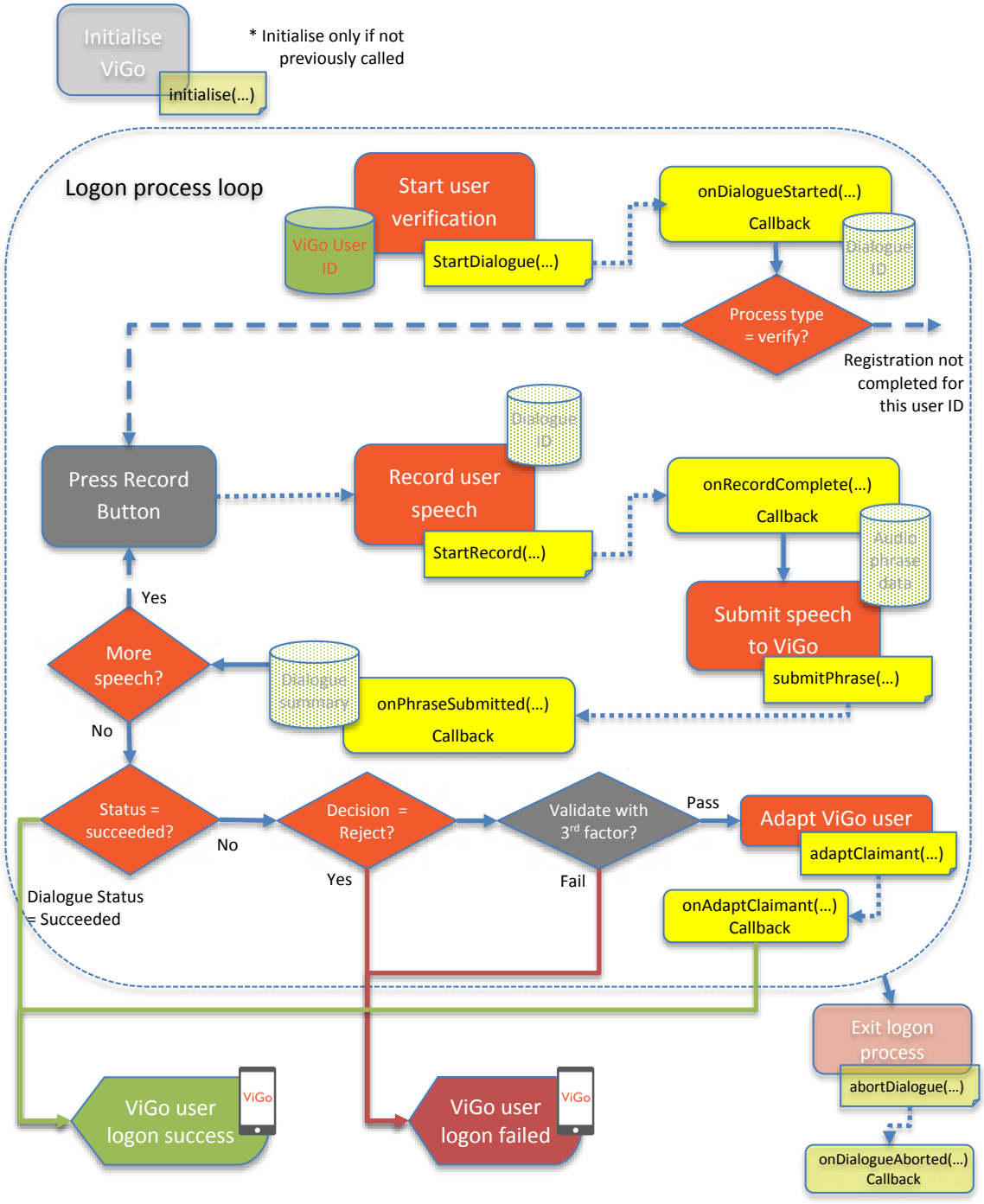
User voice registration process

The user voice registration process is the first interaction that the user has with ViGo and may take place at any time, such as when the user first runs the application or just before they first require to voice logon. This is a decision that should be considered at design time to best fit to the application workflow and provide a good user experience.



User voice logon process

The voice logon process is the main interaction that the user of the app will have with the ViGo system. It is used at the point that your application needs to validate that the registered user is who they claim to be, so that their access to your system or secured parts of your application can be authorized.



ViGo for iOS

Development of ViGo-enabled apps on iOS is fast and straightforward, and just requires a developer to include the relevant ViGo for iOS library, import the header, and include the supplied code snippets in the correct place in the application.

The ViGo.h header file defines the interface for ViGo, and includes all the functions required to enable voice biometrics in your application. A sample project for a small simple iOS app is included with the source, and demonstrates how extremely easy it is to implement basic ViGo functionality in a matter of hours.

iOS Interface

Initialize ViGo application

```
@interface ViGo : NSObject
+(ViGo *)shared;
-(void)initCredential:(NSString*)url andUsername:(NSString*)userName andPassword:(NSString*)passWord
andOrgId:(NSString*)orgID;
-(NSString*)getUrl;
-(NSString*)getUserName;
-(NSString*)getPassword;
-(NSString*)getOrgID;
-(NSDictionary*)getRecordSettings;
@end
```

Must be called before using any ViGo functions. It initializes the library using the ViGo application credentials which you received when you registered for ViGo (see pre-requisites section above).

Voice registering new ViGo user

```
@interface RegisterClaimantClass : NSObject <NSXMLParserDelegate>
- (void)onClaimantRegistered:(void (^)(NSDictionary* dict, NSError *error))_completionBlock;
@end
```

Every new ViGo user must be voice registered with the service, and this is achieved using RegisterClaimant (a Claimant is just another name for a single ViGo user). It is

important to note that each call of this function will increase your ViGo user count, so it is essential that any errors or aborted registrations are correctly managed using the DeleteClaimant method, and any open dialogues are also aborted (see below).

The claimantID returned from RegisterClaimant in the callback specifically and uniquely represents a single ViGo user, and must be persisted by your application and used for all interactions with that user.

Delete user

```
@interface DeleteClaimantClass : NSObject <NSXMLParserDelegate>
- (id)initWithData:(NSString*)claimantID;
- (void)onClaimantDeleted:(void (^)(NSDictionary* dict, NSError *error))_completionBlock;
@end
```

Should be used only to clean up after an error during voice registration, or if the user aborts the process before it is successfully completed.

It is important to note that calling this method with an active voice registered ViGo user will unrecoverably delete them from system, and they will no longer be able to interact to voice logon.

Start new interaction with ViGo user

```
@interface StartDialogueClass : NSObject <NSXMLParserDelegate>
- (id)initWithData:(NSString*)claimantID andConfigID:(NSString*)configurationID andCallRef:(NSString*)callReference;
- (void)onDialogueStarted:(void (^)(NSDictionary* dict, NSError *error))_completionBlock;
@end
```

An interaction with a user, such as voice registration and logon, is known as a *dialogue*. Before any activity in ViGo it is necessary to start a dialogue for the required ViGo user, using the StartDialogue function. The user's unique ID (claimantID) obtained from RegisterClaimant is passed in as a parameter.

The dialogueID returned in the callback should be persisted and used in the current voice registration or logon process as appropriate. It is automatically closed when the ViGo workflow for the interaction completes, or must be aborted on error or process exit.

For each interaction there is also a requirement to identify the voice biometric parameters that should be used by the ViGo system. This is achieved by passing in the

correct configuration identifier. A number of different identifiers are provided when you register for ViGo, and specifically determine the phrase that will be prompted for the user to say.

It is extremely important that the correct configuration identifier be used for a ViGo user and that it is identical to the one used when they registered, or that user will not be able to voice logon to the system.

Abort interaction with user

```
@interface AbortDialogueClass : NSObject <NSXMLParserDelegate>
- (id)initWithData:(NSString*)dialogueID;
- (void)onDialogueAborted:(void (^)(NSDictionary* dict, NSError *error))_completionBlock;
@end
```

Should be used to exit any interaction currently in progress with a user, such as if the user aborts a voice registration or an error occurs that prevents the interaction completing.

Only one voice registration dialogue can be active for any given claimant, so it is important to abort any open dialogues that do not fully complete, or it will potentially block further voice registrations for the user for up to an hour. Should be combined with DeleteClaimant when a registration is aborted to prevent 'tombstoned' users.

Submit speech audio for a user

```
@interface SubmitPhraseClass : NSObject <NSXMLParserDelegate>
- (id)initWithData:(NSString*)dialogueID andUtterance:(NSData*)audioData andPrompt:(NSString*)promptHint;
- (void)onPhraseSubmitted:(void (^)(NSDictionary* dict, NSError *error))_completionBlock;
@end
```

Used to submit the recorded speech during an interaction with the user, such as during voice registration or logon. The ViGo system will keep requesting speech from the user until it has enough to complete a voice registration, typically at least 3-4 times, or can accurately verify the user during a voice logon process, typically only 1-2 times.

If the speech submitted is not of sufficient quality, such as if the environment is noisy or the user has said the phrase incorrectly, then additional speech may be requested up to a maximum number of times when either a dialogue success or failure status will be returned.

The speech submitted must be in the correct format, and a helper function for recording audio is provided to simplify the process. If you want to develop or use your own audio recording functionality then please refer to the ViGo REST API guide for further details on the formats accepted by the ViGo system.

Adapt user speech model

```
@interface AdaptClaimantClass : NSObject <NSXMLParserDelegate>
- (id)initWithData:(NSString*)claimantID andDialogueID:(NSString*)dialogueID;
- (void)onClaimantAdapted:(void (^)(NSDictionary* dict, NSError *error))_completionBlock;
@end
```

Adaptation of the user speech model is an important and mandatory function provided by ViGo, and is essential for maintaining security and accuracy over time for all your users. As described above, one form of adaptation is implemented automatically on the ViGo server, and this must be used in conjunction with supervised adaptation implemented on the client-side using the AdaptClaimant function.

It is very important that supervised adaptation is used in conjunction with a third-factor, such as a pin or swipe that is collected from the user during voice registration, or based on existing personal information that your application has already persisted and could be used to validate user identity. This additional factor is used to authenticate the user when the ViGo system has been unable to validate a user and return an *accept* status, but has also not determined that the user is a *strong reject* where they are highly likely to be an imposter.

Note that supervised adaptation will typically only happen once or twice shortly after voice registration, and after calling AdaptClaimant a few times for a user then it will rarely be required again unless perhaps the user is in a difficult environment.

It is extremely important that this function only be called after a user is validated using a third factor, as demonstrated in the ViGo demo app.

Audio recording

```
@interface ViGoAudioRecorder : AVAudioRecorder
- (void)onRecordComplete:(float)time andCallBack:(void (^)(NSData* audioData, NSError *error))_completionBlock;
@end
```

This helper function records a fixed duration of speech from the device microphone, in a suitable format for submission to the ViGo system. Refer to the ViGo REST API documentation if you would like to know more details on the formats accepted.

Building your app

As soon as you have determined where in your application workflow you will put the voice registration and logon process, you will use these functions and their associated callbacks to build ViGo functionality in to your app.

Refer to the source snippets supplied with the ViGo iOS library in conjunction with the process workflow above for detail of how these functions are typically used in an application. The VoiceVault Vigo demo app (available as Android and iOS versions) additionally allows you to experience the ViGo convenience and ease of use, and also demonstrates best practices in UX and UI design for voice biometric applications.