Equity and Access to Archives through Digital Library Translation and Transcription Services

Florida International University - Final Report

EXECUTIVE SUMMARY

Through the generous support of the Society of American Archivists Foundation, the Florida International University (FIU) Libraries have been able to explore the development of technology for archives in order to provide archive patrons with increased multilingual and multiformat access to digital collections materials. The goal was to provide automated translation for materials written or spoken in English for non-English speaking patrons, and vice-versa for content written or spoken in other languages. Furthermore, automated transcription and text-to-speech would provide improved access for individuals with visual or hearing impairments.

The development team at FIU conducted a study of existing Microsoft APIs in the areas of 1) translation; 2) speech-to-text; and 3) text to speech, with five testing scenarios outlined in the report below. Results of this testing highlighted both the possibilities and limitations of the Microsoft APIs. Due to the limitations of the technology, it is likely that archives will need inhouse technical support and/or development experience to implement these transcription and translation functions. Additionally, automation is limited. This means that materials will need to be transcribed or translated in batches, rather than “on-the-fly” as we had originally intended. However, this testing was still fruitful and may be used by archives to broaden their reach and strengthen connections with patrons by expanding the types of services provided. These services may include translation of texts for patrons upon request, speech-to-text transcription for hearing impaired patrons, as well as text-to-speech for visually impaired patrons. This technology may also be used by archives to enhance understanding and organization of their collections, particularly in instances where it is necessary to describe lengthy audio recordings that have not been manually transcribed or in instances where the archive does not have personnel with the language expertise it may need to process and describe a collection. In short, this technology can save archives time, allowing archivists to focus their energies in other areas.

Since we were not able to automate the process, we do not have plans to share the results of this round of development broadly through publication. However, the technical report and code are freely available through the GitHub repository for others to use and develop further. The FIU team plans on applying for additional funding to resolve current obstacles (including the need for additional computational power), to enhance our processes, and refine the code. The next phase of our development will focus on automation of text translation and incorporation into our digital library system, allowing patrons to request translations of text on-the-fly. The FIU team also plans on convening local focus groups to gather information about specific projects that may benefit from this new service. Two projects that may have immediate application are FIU’s recently funded Collections as Data grant for *dLOC as Data* and the *Digital Archive of Religion in Latin America and the Caribbean*.

BACKGROUND

The Florida International University (FIU) Libraries led the testing and exploration of translation and transcription functionality into the open source digital repository systems, dPanther and dLOC. Upon embarking on this project in 2018, the intended outcome was increased access to digital collections materials written or spoken in English for non-English speaking patrons and vice-versa for content written or spoken in other languages, as well as providing additional measures of access for individuals with visual or hearing impairments.

To accomplish this goal, FIU has leveraged existing technologies made available through Microsoft’s Azure Cognitive Services APIs (<https://azure.microsoft.com/en-us/services/cognitive-services/>). The Microsoft services include 63 languages supported for translation, 18 for transliteration, and 51 for dictionary, as well as speech to text and text to speech functionality. As a current Microsoft client, FIU was able to test these API functions and investigate their integration into the open source digital repository systems, which are built upon the SobekCM platform.

TECHNICAL METHODOLOGY

In this study, we aimed to test the Microsoft Translator APIs within the dPanther system. Since the dPanther system is hosted and developed within the FIU library internal network, utilizing a Microsoft .NET framework, both REST APIs and SDK were implemented for the experiments. The experimentation environment consisted of a blended in-house hosted Microsoft .NET 4.5 Framework and Microsoft Cognitive Service hosted in the Cloud. Our targets for the study involved two aspects: Microsoft Azure Translator APIs for textual materials in dPanther and the Microsoft Azure Speech Service for the multi-media assets in dPanther. Since dPanther is developed based on an open source repository, SobekCM, which is also developed under the Microsoft .NET Framework, the integration with the Microsoft Azure Translator service was realized by connecting an additional .NET 4.5 project into the SobekCM solution.

We proposed two separate workflows for the Microsoft Azure Translator APIs for textual materials in dPanther and the Microsoft Azure Speech Service for the multi-media assets respectively. For the Microsoft Translator Text APIs in dPanther, we proposed a purely front-end solution by using HTML/JavaScript and Restful APIs from Microsoft Azure.

In Figure 1, we illustrate the workflow for Microsoft Translator Text APIs in dPanther, broken out into 3 steps:

1. Capture the dPanther textual documents into a text stream by using JavaScript.
2. Request translation and/or sentence from Translator APIs by calling the Restful APIs using JavaScript.
3. Parse the response from the Translator APIs and display it in HTML format. The major benefit of this solution is that it will not cause much backend change and it is very flexible to scale up with future backend changes.

A screenshot of a cell phone

Description automatically generatedFigure 1. Workflow of Microsoft Translator Text APIs in dPanther

For the speech service, we developed a backend C# SDK project. In Figure 2, we illustrate the workflow as follows:

1. Form the input: Since the Cognitive Service only handles .wav files, the first step is to convert any media file into .wav format. In our experiment, we use a python script to convert a YouTube URL into a .wav file.
2. Convert .wav audio source into text: By using the Azure Speech Service C# SDK, we can convert the audio input from step 1 into text format by calling the APIs within the C# project. The output will be saved as a text transcript in the document’s original language.
3. Translate into target language: By using the Azure Speech Service C# SDK, we can translate the text materials from step 2 into any supported language in text format. The output will be saved as a transcript in the target language.
4. Convert the translated text into audio: By using the Azure Speech Service C# SDK, we can convert the translated text in step 3 into a .wav format by calling the APIs.
5. Upload into potential media server: This step is not going to be covered in this study since every organization handles their media files differently (e.g. from YouTube to dedicates Media servers). However, this should not be a critical task since most media servers have their own programming interface for developers to upload video content programmatically.

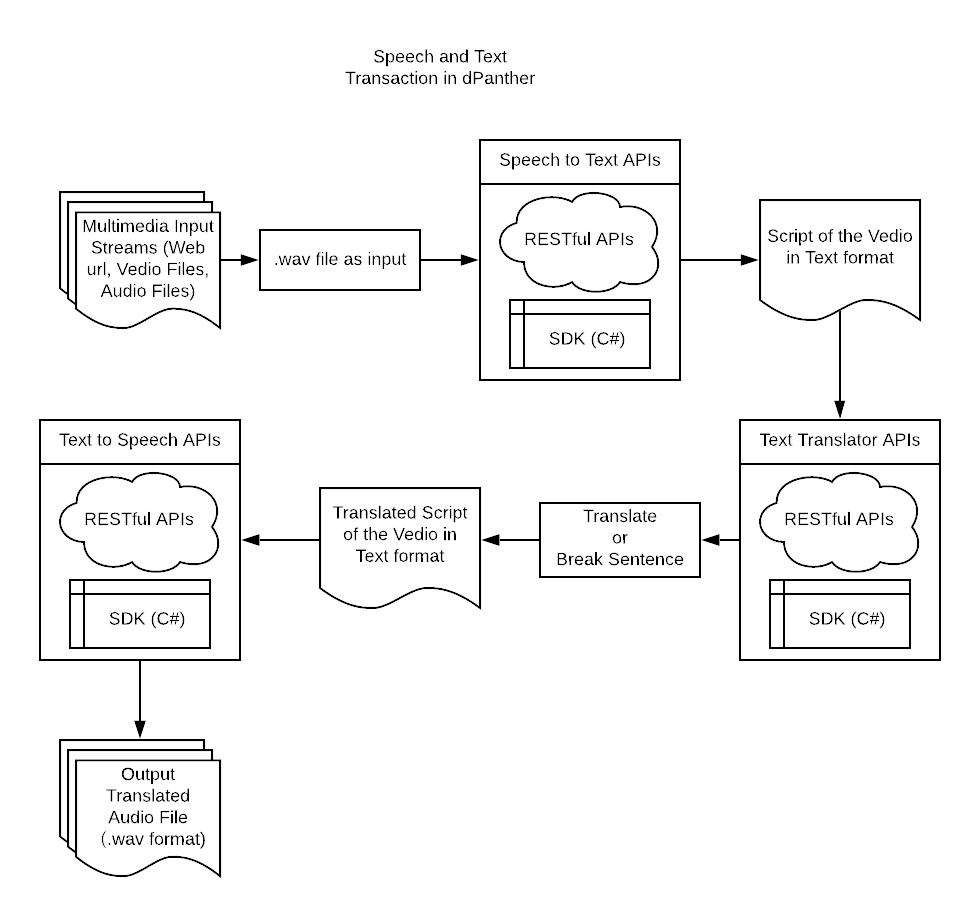


Figure 2. Workflow of Speech and Text Translation in dPanther

SUMMARY OF RESULTS

We have summarized our results in two parts: 1) Microsoft Azure Translator APIs for textual materials in dPanther and 2) Microsoft Azure Speech Service for the multi-media assets in dPanther. The summarization focuses on the interactions between the in-house application and the Azure cloud APIs. The evaluation provides details regarding the performance and limitations of each solution.

Microsoft Azure Translator APIs

For the Microsoft Azure Translator APIs for textual materials in dPanther, two APIs calls are used: Translate and Break Sentence. We design our experiment into the following steps:

1. Extract one of the PDF files into .txt format and use it as the source input: <http://dpanther.fiu.edu/sobek/content/FI/12/09/03/22/00001/FI12090322_pdf.txt>
2. Use this source text as input to test the performance of the translate API

### Translate APIs

**Performance:** Translation directly from raw text files is poor. The first problem is that the raw file from the PDF conversion does not maintain formatting, often contains many illegal/invalid characters, and will cause the API call to fail. We randomly chose one page, page 10 for this study specifically, that contains the textual materials only (no images). After we have removed illegal characters, the content was successfully translated into both Chinese and Spanish. For 2,863 characters, is takes around 0.3 seconds to finish the translation.

**Limitations:** The Translator Text API v3.0 provides a modern JSON-based Web API (<https://docs.microsoft.com/en-us/azure/cognitive-services/translator/reference/v3-0-reference>). We need convert our input files to a JSON array. The array can have 100 elements at most. The entire text included in the request cannot exceed 5,000 characters including spaces. (<https://docs.microsoft.com/en-us/azure/cognitive-services/translator/reference/v3-0-translate>)

### Break Sentence APIs

**Performance:** The test results for the Break Sentence APIs turned out to be very positive and by following the return position, we are able to separate the sentences clearly. However, this API is only working with well formatted paragraphs. It cannot automatically identify the sentences if the format is lost. Therefore, if the input source is a text file directly converted from a PDF file, the API does not working appropriately.

**Limitations:** For the Break Sentence API, it requests the body with a JSON array. The array can have 100 elements at most. The text value of an array element cannot exceed 10,000 characters, including spaces. The entire text included in the request cannot exceed 50,000 characters, including spaces. Also, if the language query parameter is specified, then all array elements must be in the same language. Otherwise, language auto-detection is applied to array element independently (<https://docs.microsoft.com/en-us/azure/cognitive-services/translator/reference/v3-0-break-sentence>).

Microsoft Azure Speech Service

For the Microsoft Azure Speech Service for the multi-media assetsin dPanther, three APIs are used: Speech to Text, Translate (SDK), and Text to Speech. We design our experiment into the following steps:

1. Extract three of the mp3 files into .wav format and use them as the source input:

* <http://dpanther.fiu.edu/dpanther/items/itemdetail?bibid=FI07040607&vid=00001>
* <http://dpanther.fiu.edu/dpanther/items/itemdetail?bibid=FI07072121&vid=00001>
* <http://dpanther.fiu.edu/dpanther/items/itemdetail?bibid=FI14090734&vid=00001>

1. Use these source audio files as input to test the performance of the speech-to-text APIs, and output text.
2. Use output text as input to test translate APIs with Spanish output .txt files.
3. Use output .txt files as input to text the performance text-to-speech APIs, and output with Spanish audio files.

### Speech-to-Text APIs

**Performance:** The Speech SDK cannot recognize speech from the raw .mp3 files, because the Speech SDK supports WAV/PCM 16-bit, 16KHz/8KHz, single-channel audio for speech recognition, but additional audio formats are supported for a speech-to-text REST endpoint. Also, after recognizing the entire audio file, Speech SDK only returns the first spoken statement or otherwise cannot be recognized. The problem is caused by the SDK’s limitation to only recognize when the end of a single statement is determined by listening for silence at the end or until maximum of 15 seconds of audio is processed (<https://docs.microsoft.com/en-us/dotnet/api/microsoft.cognitiveservices.speech.speechrecognizer.recognizeonceasync?view=azure-dotnet>).

We spliced 3 target input .mp3 files to 95 .wav files with single statements. The performance is not very good, some audio files can be recognized, and others cannot be recognized very well. Also, a small part of audio file cannot be recognized. For the recognize time, it depends how long the speech can be recognized. For example, a 10 second file (Test3-10s10) can be recognized for about 6 seconds.

**Limitations:** In this case, speech-to-text APIs only can return results by listening for silence; a single utterance/statement or maximum of 15 seconds of audio speech.

### Translate  APIs

**Performance:** The translate from recognized English to Spanish is good. It can translate from the default language to the target language directly. For example, a 10 second file (Test3-10s10) with English can be recognized with 30 characters, and can be translated and output to a Spanish file in 2 seconds.

### Text-to-Speech APIs

**Performance:** Text-to-speech using Speech SDK or REST API to convert. It supports standard, neural, or custom voices (<https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/text-to-speech>). The performance for testing is good. For example, after a 10 second file (Test-10s10) is recognized with output to a Spanish file, the text-to-speech APIs will generate an audio file spoken in Spanish within 2 seconds.

**Limitations:** Text-to-Speech only provides support to convert speech in files that are under 10 minutes. For speech that is longer than 10 minutes, files need use the Long Audio API. However, the Long Audio API doesn’t like synthesis performed using the Speech SDK, the responses aren’t returned in real time, the synthesized audio is downloaded when made available from the service, and the Long Audio API only supports neural voices. For neural voices, Microsoft speech service only supports German, English, Italian and Chinese at this time. (<https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/long-audio-api>)

PROJECT DELIVERABLES

* 1. A detailed technical report outlining the methodology for testing the incorporation of Microsoft’s Translation Services into the open source dPanther/SobekCM system is available at (<https://github.com/zhongzhou1/dPantherAzureTranslate>).
  2. Open source code, applicable to other organizations who use the SobekCM platform, along with detailed code documentation (see Appendix).
  3. Links to sample transcription and translation services applied to content in the dPanther archives (<https://github.com/zhongzhou1/dPantherAzureTranslate>).

SUMMARY OF GRANT EXPENSES

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Hours** | **Cost** | **Funding Source** |
| Graduate Developer | 300 hours | $5,000 | SAA |
| Lead Developer | 90 hours | $4,500 | FIU (cost share) |
| Project Management | 10 hours | $500 | FIU (cost share) |
| Microsoft Azure Translation Services Testing of API |  | Included in FIU’s MS client license | FIU (cost share) |
| **TOTAL** |  | **$10,000** |  |

TIMELINE

|  |  |
| --- | --- |
| July 2018 | Award distributed |
| August 2018 | Developer position posted and hired |
| September 2018 – May 2019 | Microsoft Azure Translator APIs testing for textual materials in dPanther |
| May 2019 | No Cost Extension Granted |
| May 2019 – November 2019 | Microsoft Azure Speech Service testing for the multi-media assets in dPanther |
| November 2019 | Apply Microsoft API Services to sample content |
| December 2019 | Code documentation and GitHub dissemination |
| December 2019 | Final Report Submitted |

APPENDIX

**Technical Documentation**

Function Analysis

In this section, we summarize the functionality Translator Text APIs and Speech APIs.

**Translator Text APIs**

Supported Language

Microsoft still grows the pool of supported languages. By the time of this study, in API version 3, there are 63 languages supported to translation, 18 for transliteration, and 51 for dictionary. The best way to check the supported languages of Translator Text APIs is by querying this RESTful API end point: <https://api.cognitive.microsofttranslator.com/languages?api-version=3.0>

Translate

Function Summary

Translate the text

Implementation

API end point: <https://api.cognitive.microsofttranslator.com/translate?api-version=3.0>

Supported operation: POST

Query parameters:

|  |  |  |
| --- | --- | --- |
| Name | Description | Required? |
| api-version | Version of the API requested by the client. Value must be 3.0. | Yes |
| to | Specifies the language of the output text. The target language must be one of the supported languages included in the translation scope. For example, use to=de to translate to German.  *It's possible to translate to multiple languages simultaneously by repeating the parameter in the query string. For example, use to=de&to=it to translate to German and Italian.* | Yes |
| from | Defines whether the text being translated is plain text or HTML text. Any HTML needs to be a well-formed, complete element. Possible values are: plain (default) or html. | No |
| textType | Defines whether the text being translated is plain text or HTML text. Any HTML needs to be a well-formed, complete element. Possible values are: plain (default) or html. | No |
| category | A string specifying the category (domain) of the translation. This parameter is used to get translations from a customized system built with [Custom Translator](https://docs.microsoft.com/en-us/azure/cognitive-services/translator/customization). Add the Category ID from your Custom Translator [project details](https://docs.microsoft.com/azure/cognitive-services/translator/custom-translator/how-to-create-project#view-project-details) to this parameter to use your deployed customized system. Default value is: general. | No |
| profanityAction | Specifies how profanities should be treated in translations. Possible values are: NoAction (default), Marked or Deleted. To understand ways to treat profanity, see [Profanity handling](https://docs.microsoft.com/en-us/azure/cognitive-services/translator/reference/v3-0-translate#handle-profanity). | No |
| profanityMarker | Specifies how profanities should be marked in translations. Possible values are: Asterisk (default) or Tag. To understand ways to treat profanity, see [Profanity handling](https://docs.microsoft.com/en-us/azure/cognitive-services/translator/reference/v3-0-translate#handle-profanity). | No |
| includeAlignment | Specifies whether to include alignment projection from source text to translated text. Possible values are: true or false (default). | No |
| includeSentenceLength | Specifies whether to include sentence boundaries for the input text and the translated text. Possible values are: true or false (default). | No |
| suggestedFrom | Specifies a fallback language if the language of the input text can't be identified. Language auto-detection is applied when the from parameter is omitted. If detection fails, the suggestedFrom language will be assumed. | No |
| fromScript | Specifies the script of the input text. | No |
| toScript | Specifies the script of the translated text. | No |
| allowFallback | Specifies that the service is allowed to fallback to a general system when a custom system does not exist. Possible values are: true (default) or false.    allowFallback=false specifies that the translation should only use systems trained for the category specified by the request. If a translation for language X to language Y requires chaining through a pivot language E, then all the systems in the chain (X->E and E->Y) will need to be custom and have the same category. If no system is found with the specific category, the request will return a 400 status code. allowFallback=true specifies that the service is allowed to fallback to a general system when a custom system does not exist. | No |

Related to dPanther

By utilizing this service, we can dynamically translate the language purely from the front end without any additional server-end setup.

Transliterate

Function Summary

Converts text in one script of one language to another script.

Implementation

API end point: <https://api.cognitive.microsofttranslator.com/transliterate?api-version=3.0>

Supported operation: POST

Query parameters:

|  |  |  |
| --- | --- | --- |
| Name | Description | Required? |
| api-version | Version of the API requested by the client. Value must be `3.0`. | Yes |
| language | Specifies the language of the text to convert from one script to another. Possible languages are listed in the `transliteration` scope obtained by querying the service for its [supported languages](./v3-0-languages.md). | Yes |
| fromScript | Specifies the script used by the input text. Look up [supported languages](./v3-0-languages.md) using the `transliteration` scope, to find input scripts available for the selected language. | Yes |
| toScript | Specifies the output script. Look up [supported languages](./v3-0-languages.md) using the `transliteration` scope, to find output scripts available for the selected combination of input language and input script. | Yes |

Related to dPanther

dPanther will not implement this function

Detect

Function Summary

Identifies a piece of text language.

Implementation

API end point: <https://api.cognitive.microsofttranslator.com/detect?api-version=3.0>

Supported operation: POST

Query parameters:

|  |  |  |
| --- | --- | --- |
| Name | Description | Required? |
| api-version | Version of the API requested by the client. Value must be `3.0`. | Yes |

Related to dPanther

By utilizing this function, dPanther system will be able to automatically detect the language of the asset from the documents or scripts.

Break Sentence

Function Summary

Identifies the boundary position of the sentence in the text.

Implementation

API end point: https://api.cognitive.microsofttranslator.com/breaksentence?api-version=3.0

Supported operation: POST

Query parameters:

|  |  |  |
| --- | --- | --- |
| Name | Description | Required? |
| api-version | Version of the API requested by the client. Value must be `3.0`. | Yes |
| language | Language tag identifying the language of the input text. If a code is not specified, automatic language detection will be applied. | Optional |
| script | Script tag identifying the script used by the input text. If a script is not specified, the default script of the language will be assumed. | Optional |

Related to dPanther

There are many documents script from dPanther are converted from PDF and thus, the sentences are not probably separated. By utilizing this function, dPanther

Dictionary

Function Summary

Provides alternative translations for a word and a small number of idiomatic phrases.

Implementation

API end point: <https://api.cognitive.microsofttranslator.com/dictionary/lookup?api-version=3.0>

Supported operation: POST

Query parameters:

|  |  |  |
| --- | --- | --- |
| Name | Description | Required? |
| api-version | Version of the API requested by the client. Value must be `3.0`. | Yes |
| from | Specifies the language of the input text. The source language must be one of the [supported languages](./v3-0-languages.md) included in the `dictionary` scope. | Yes |
| to | Specifies the language of the output text. The target language must be one of the [supported languages](./v3-0-languages.md) included in the `dictionary` scope. | Yes |

Related to dPanther

Speech Service

Speech-to-text

Function Summary

Speech-to-text, enables real-time translation from audio streams to text with Microsoft Azure Speech Service. Speech SDK and REST APIs can recognize audio from a microphone, stream or audio files from storage. The SDK supports WAV/PCM 16-bit, 16 kHz/8kHz, single-channel audio.

Implementation

The features of Speech-to-text are available in two forms: SDK or REST APIs.

|  |  |  |
| --- | --- | --- |
| **Use case** | **SDK** | **REST** |
| Transcribe short utterances (<15 seconds). Only supports one final transcription result. | Yes | Yes\* |
| Continuous transcription of long utterances and streaming audio (>15 seconds). Supports interim and final transcription results. | Yes | No |
| Derive intents from recognition results with LUIS. | Yes | No\*\* |
| Batch transcription of audio files asynchronously. | No | Yes\*\*\* |
| Create and manage speech models. | No | Yes\*\*\* |
| Create and manage custom model deployments. | No | Yes\*\*\* |
| Create accuracy tests to measure the accuracy of the baseline model versus custom models. | No | Yes\*\*\* |
| Manage subscriptions. | No | Yes\*\*\* |

*\*Using the REST functionality you can transfer up to 60 seconds of audio and will receive one final transcription result.*

*\*\*LUIS intents and entities can be derived using a separate LUIS subscription. With this subscription, the SDK calls LUIS for you and provide entity and intent results. With the REST API, you call LUIS yourself to derive intents and entities with your LUIS subscription.*

*\*\*\*These services are available using the cris.ai endpoint. See Swagger reference.*

Related to dPanther

Text-to-speech

Function Summary

Text-to-speech can convert text into natural human-like synthesized speech.

Implementation

The features of Text-to-speech are available in two forms: SDK or REST APIs.

|  |  |  |
| --- | --- | --- |
| Use case | SDK | REST |
| Convert text to speech | Yes | Yes |
| Upload datasets for voice adaptation | No | Yes\* |
| Create and manage voice font models | No | Yes\* |
| Create and manage voice font deployment | No | Yes\* |
| Create and manage voice font tests | No | Yes\* |
| Manage subscriptions | No | Yes\* |

Related to dPanther

Intent recognition

Function Summary

Intent can help users to use terms feel natural.

Implementation

LUIS integrates with the Speech service to recognize intents from speech.

|  |  |
| --- | --- |
| **Key type** | **Purpose** |
| Authoring | Lets you create and modify LUIS apps programmatically |
| Starter | Lets you test your LUIS application using text only |
| Endpoint | Authorizes access to a particular LUIS app |

Related to dPanther

Speech translation

Function Summary

Speech translation can translate with real-time, multi-language speech-to-speech and speech-to-text translation with audio streams

Implementation

Here are the features available via the Speech SDK and REST APIs

|  |  |  |
| --- | --- | --- |
| Use Case | SDK | REST |
| Speech-to-text translation with recognition results. | Yes | No |
| Speech-to-speech translation. | Yes | No |
| Interim recognition and translation results. | Yes | No |

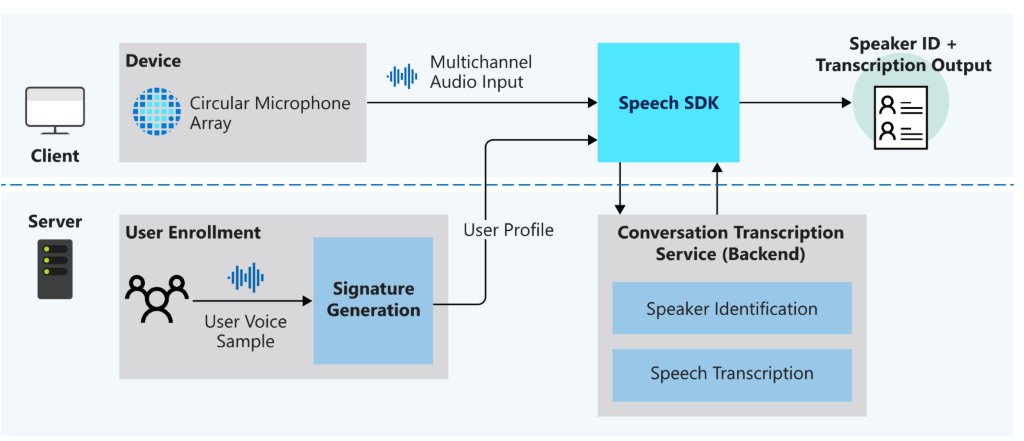
Related to dPanther

Conversation transcription

Function Summary

Conversation Transcription is a speech-to-text solution that can determine who said what and when.

Implementation



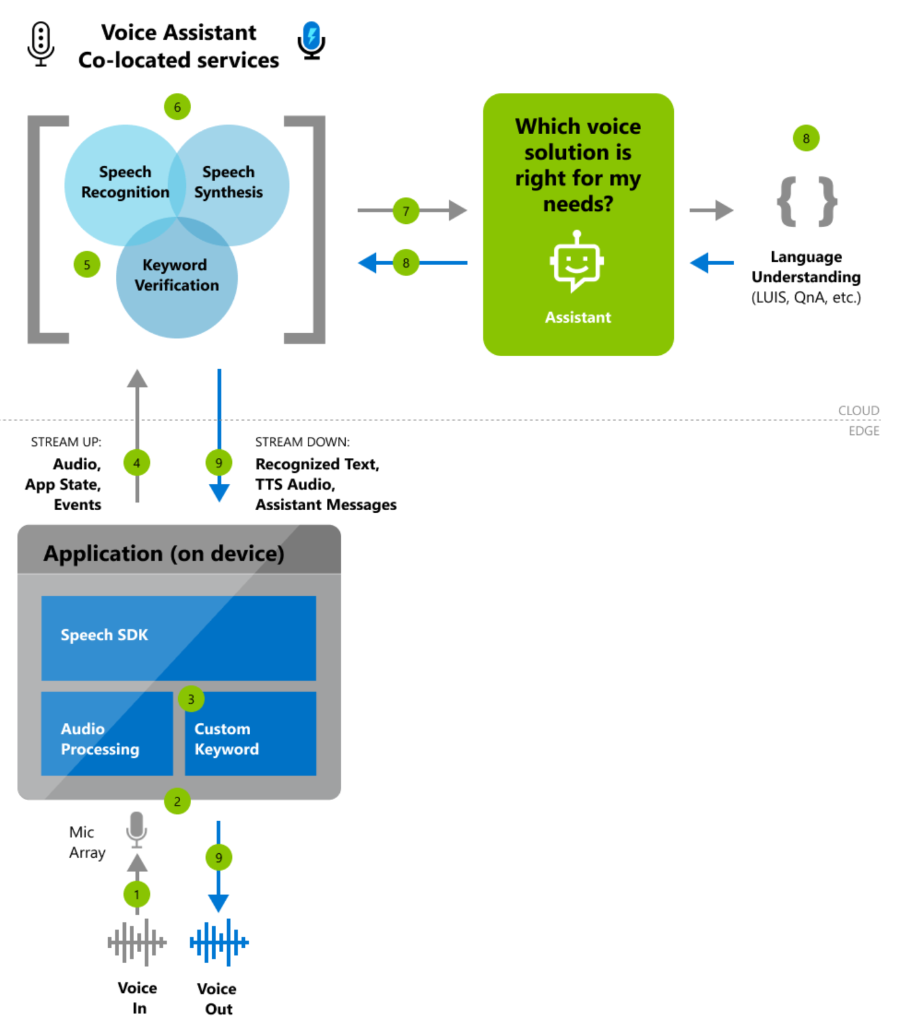
|  |  |
| --- | --- |
| Timestamps | each speaker utterance has a timestamp, so that you can easily find when a phrase was said. |
| Readable transcripts | transcripts have formatting and punctuation added automatically to ensure the text closely matches what was being said. |
| User profiles | user profiles are generated by collecting user voice samples and sending them to signature generation. |
| Speaker identification | speakers are identified using user profiles and a *speaker identifier* is assigned to each. |
| Multi-speaker diarization | determine who said what by synthesizing the audio stream with each speaker identifier. |
| Real-time transcription | provide live transcripts of who is saying what and when while the conversation is happening. |
| asynchronous transcription | provide transcripts with higher accuracy by using a multichannel audio stream. |

Related to dPanther

Voice assistants

Function Summary

Voice assistants using the Speech service to create natural, human-like conversational interfaces for applications and experiences.



Implementation

|  |  |
| --- | --- |
| **Category** | **Features** |
| [Custom keyword](https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/speech-devices-sdk-create-kws) | Users can start conversations with assistants with a custom keyword like “Hey Contoso.” An app does this with a custom keyword engine in the Speech SDK, which can be configured with a custom keyword [that you can generate here](https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/speech-devices-sdk-create-kws). Voice assistants can use service-side keyword verification to improve the accuracy of the keyword activation (versus the device alone). |
| [Speech to text](https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/speech-to-text) | Voice assistants convert real-time audio into recognized text using [Speech-to-text](https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/speech-to-text) from the Speech service. This text is available, as it's transcribed, to both your assistant implementation and your client application. |
| [Text to speech](https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/text-to-speech) | Textual responses from your assistant are synthesized using [Text-to-speech](https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/text-to-speech) from the Speech service. This synthesis is then made available to your client application as an audio stream. Microsoft offers the ability to build your own custom, high-quality Neural TTS voice that gives a voice to your brand. To learn more, [contact us](mailto:mstts@microsoft.com). |

Related to dPanther