# A Voice-controlled Streaming Jukebox based on IBM Bluemix Cloud Service

0116314 Li-An Yang
Department of Computer Science
National Chiao Tuing University, Hsinchu, Taiwan (R.O.C)

***Abstract*—In this proposal, we would explain our idea that merges music streaming service into Raspberry Pi to construct a cloud streaming server. Compared with a similar product Pi Musicbox that also stream music by Mopidy, this project also supports voice control and other fantastic features with the help from cloud PaaS IBM Bluemix and Watson, such as choosing preferred music genres or playlists according to present mood via cognitive services provided by Watson API. In this project, we follow the trend of IoT and take advantage of the cloud computing service to promote this new era of jukebox.**

***Keywords—Spotify, Mopidy, Raspberry Pi, IBM Bluemix, Watson***

## I. Introduction

Nowadays, with the networks existing almost everywhere, not only technical devices but also common appliances like light bulbs and CD players could be connected to the cloud and make our life a lot more convenient.

Music streaming services are also very popular throughout mobile devices these days, yet it could be disrupted by a sudden phone call or notifications, and would reduce battery volume. After all, they are not generally designed for music display. Hence, we want to create a standalone music player for Spotify[1] that can keep streaming music under voice and HTTP/MPD[2] remote control without any disturbance. It will be designed to act and response in a smart way with the help of artificial intelligence technique from IBM Watson[3].

## II. Related Work

There are already a handful of implementations on Raspberry Pi[4] connecting streaming service as a server for mobile devices to control. For example, Pi MusicBox[5] in Fig. 1 provides makers an easy way to come up with their own streaming music player with WiFi support. Comparing our Streaming Jukebox with Pi MusicBox, there exists some similarities as well as differences in between:

### A. *Both are based on Mopidy[6]*

Mopidy is an extensible music server written in Python. We can find both design instilled with Mopidy package in Raspberry Pi to connect to local disk or radio streams. By doing so, we can then edit the music playlist in various way, which is what makes our proposed model different from current existing implementations.

### B. *Intellectual Control v.s. HTTP/MPD Remote Control*

The ability to control music smartly through interaction with the machine is definitely the most intriguing element of the Streaming Jukebox. In addition to using web or an MPD-client as an interface to remote like Pi MusicBox, thanks to Speech to Text service from IBM Watson, SJ is capable of recognizing certain human voices and converts them into texts for execution. Furthermore, it also features a user response system attached with emotion detection to help find out your personal customized playlist. These Watson services helps simplify the process and diversify the options.

Fig. 1 A DIY Project based on Pi MusixBox[4]

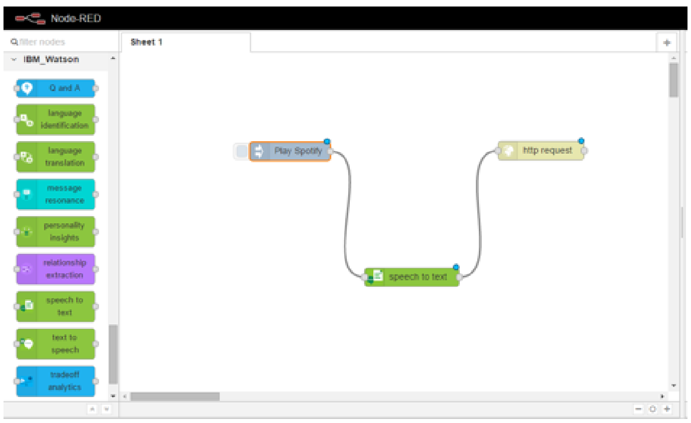

Fig. 2 Node-Red Interface from IBM Bluemix

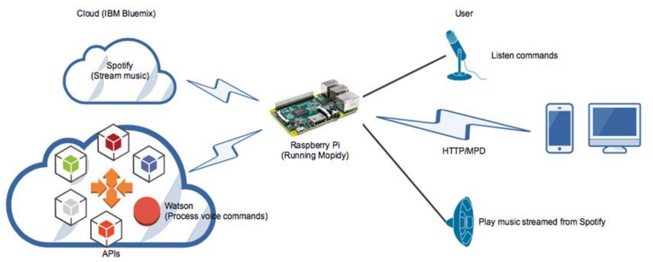

Fig. 3 The proposed model

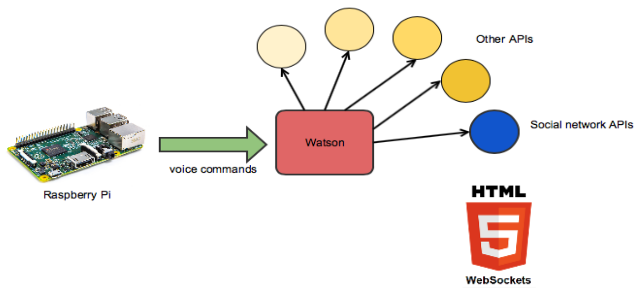

Fig. 4 Cloud computing flow

*Compatibility with APIs*

Another great advantage that outperforms other similar works is the relatively easy and clear API bridging interface provided by IBM Bluemix[7]. As shown in Fig. 2, Node-RED provides diagrams to help Bluemix users wire together APIs in straightforward ways. We will deploy not only Watson but also associating other interesting APIs such as Facebook to construct a real cloud jukebox; thus, people can share their desired music to friends easily via online social networking services. Additionally, Yahoo weather API could be involved to provide users with more friendly options to help find correspondent playlist by instantaneous weather information no matter where they are.

## III. Proposed Cloud Computing Model

The model, illustrated in Fig. 3, includes an extended single-board computer Raspberry Pi, as the jukebox, and two cloud computing instances provided by IBM Bluemix. One of the cloud instance runs Spotify for streaming music; the other instance processes voice commands made by users and instructs the jukebox to make expected actions. The jukebox is also equipped with a microphone to accept the command which will be sent to the second cloud instance and a speaker to play music streamed.

As an IoT device, Raspberry Pi is capable of utilizing Internet services through WiFi connection to make up for the lack of computing ability. The jukebox streams music from the cloud by Mopidy, and interacts with users via voice commands or other devices through HTTP/MPD APIs. While HTTP/MPD APIs can be handled by Mopidy itself, voice commands are sent to and processed by cloud to control Mopidy.

The cloud instance for processing voice commands takes advantage of APIs from the PaaS service of IBM Bluemix. Watson is one of these APIs that takes good care of voice commands with artificial intelligence. Depending on the response of Watson, other APIs, such as social network APIs, may also be invoked. Instructions are sent back to the websocket set up in raspberry pi to control the streaming music (Fig. 4). We have already come up with several fundamental commands driven by the human voice to operate the streaming jukebox, including basic operation such as play, pause, next, previous, mute… and so on. Instead of doing the computation on raspberry pi, we determine the actual commands on IBM bluemix to fully utilize its computation power and then transmit the command text to a websocket (Fig. 5) where it could call mopidy APIs. This way, we don't have to response anything back to the device which makes the HTTP request. In addition, we are not limited to recording our commands on raspberry pi and would reach the same effect by speaking to another intelligent device like our cell phone to remote control the jukebox. We tend to remain the computation in the cloud, allowing IBM bluemix to communicate bilaterally even if the device possesses a private IP.

We also proposed some fantastic features to enhance the functionality of jukebox in the evaluation result section. For instance, the personal recommendation for music uses some third-party APIs to detect current conditions such as weather or time. It would then call Spotify API to design intellectual playlists following and based on the result of the investigation, automatically and instantly.

## IV. Discussion

Since the interconnected structure has this project rely on many products provided by other companies, it is anticipated that some problems about these products might be encountered after this project goes into construction. The following are some assumed issues to discuss:

### A. *Raspberry Pi does not meet the minimum system requirement*

Although the experience in Pi Musicbox has persuaded us that Raspberry Pi is capable of running Mopidy properly to stream music and accept HTTP/MPD API requests, it is reported by the official of the software that Mopidy runs slowly on the device[5]. It is not ensured that the device would still be affordable after connections to other join of cloud and accessories are made. However, this problem can be resolved by migrating some portion of computing to the cloud to offload the overhead from Raspberry Pi if the problem does occur during the implementation period.

### B. *APIs provided by IBM Bluemix does not work as expect.*

Watson is indeed the most important API in this project. While its artificial intelligence for analyzing voice commands plays a big role to instruct the following APIs chained with it,

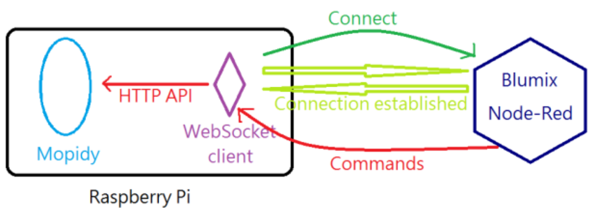

Fig. 5 The bilateral connection between WebSocket client and IBM Bluemix

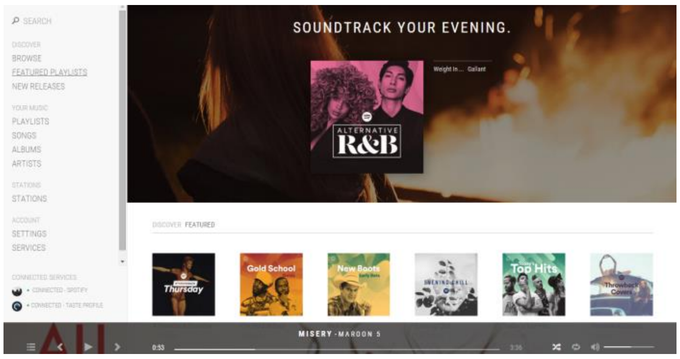

Fig. 6 A Spotify-connected web player to display information from Mopidy

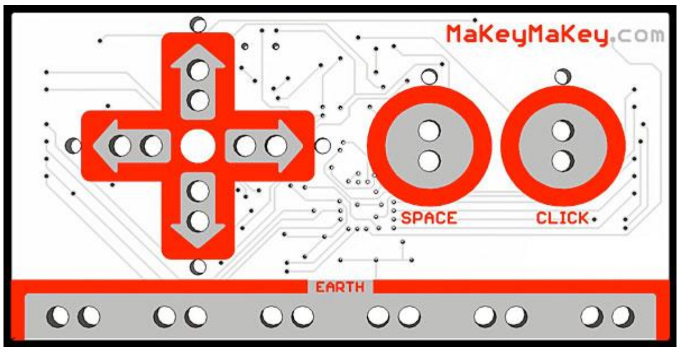

Fig. 7 The picture shows nputs with the correspondent key for detection.

and is what makes this project different from others. Despite the suspect that whether the microphone installed on Raspberry Pi can deliver the command properly without affecting the result, with a rich number of products successfully cooperated with Watson and raspberry Pi, we believe it is worth giving it a try.

## V. EVALUATION RESULTS

After several weeks of implementation, we succeed to develop a number of commands that operate the jukebox, including basic ones such as play/pause, next/previous, speech-interactive ones such as "What is the song?" and "What's next?", and finally commands that involve posting music on social media as well as carrying out personal recommendation.

### A. *Fundamental commands*

As illustrated above, basic instructions are necessary for any music players to function properly. As shown in Fig. 6, from the webpage automatically set up by the Mopidy server in raspberry pi, we can easily identify the change made by instructions such as play/pass or next/previous. What is more, we also design and connect to another innovative and plug-and-use kit called "Makey Makey [8]" as buttons (Fig. 7). It controls the streaming jukebox in a very interesting way and is very popular on Kickstarter. Refer to the reference clip and the demo video for further insight of this simplified Arduino board.

### B. *Text-to-Speech interactive commands*

Next, we move on to utilize another Watson API called Text-to-Speech to reverse information acquired from the HTTP response to audio and make SJ speaks them out. For example, it would spit out the name of the song if asked "What is the song?" Such technique could be deployed to find out the name/artist of the song or to take a glance at the play queue to know what song is ahead of us. Instead of showing the answer on the browser, we want the streaming jukebox to interact directly with the user. For now, the music will stop for a while to response the user with the correspondent answer, but a better strategy to adopt as our future work would be to let the device speak out the response while lowering the volume of the original song.

### C. *Social media and Personal recommendation*

There are several advanced features that we added on Node-RED by integrating some third-party APIs. For instance, the Facebook API is maintained to have SJ post the current playing music on Facebook immediately with Hashtag indicating the song of the name. Furthermore, we added another forecast API to detect the ongoing condition such as weather, temperatures or even the time. Consider Fig. 8 for clearer explanation, we therefore construct a preliminary version of personal recommendation based on this four types of information. SJ will choose one of the factor at random with a speech of greeting indicating the dependent factor. The factor will later serve as a keyword to search for and we would hear the recommended song out in a short time. Better seen than said, by watching the demo video will provide more understanding about these features and their workability.

## VI. FUTURE WORK

So far, we have implemented more than a few attributes, but if there is anything we can improve on, the Tone Analyzer service, one of Watson's API, will probably be included and applied to our work. To elaborate, it adopts the linguistic analysis technique to detect and interpret emotional cues, focusing on positive or negative feelings. On top of that, another intelligent component worth trying is somewhat more related to machine learning. Inspired by Spotify, we could further look at history data and try to train a customized model for each user who owns a distinctive taste of music. Such feature is already implemented and very useful on Spotify; hence, we would like to integrate it into SJ as well.

Fig. 8 Personal recommendation and its based factoral data.

On the other aspect, music recognition and lyrics synchronization are also potential for development. However, to improve the device with this kind of integration, the elevation of the quality regarding both software and hardware are necessarily required.

## VII. Conclusion

To sum up, we have introduced the concept of this sound-driven device based on IBM cloud computing service with numerous concrete examples supported by distinguished services. We hope to integrate these desired applications into one package of utilization, and will keep coming up with surprising ideas to adjust and add into Jukebox to our own desire.

## VIII.KEY REFERENCE REPORT

Mopidy Core API https://docs.mopidy.com/en/release-0.19/api/

## References

[1] Spoify API https://developer.spotify.com/web-api/

[2] MPD https://en.wikipedia.org/wiki/MPD

[3] Watson https://www.ibm.com/cloud-computing/bluemix/solutions/watson/

[4] Rasperry Pi http://www.adafruit.com/category/105

[5] Pi Musicbox http://www.pimusicbox.com/

[6] Mopidy https://www.mopidy.com/

[7] Bluemix https://console.ng.bluemix.net

[8] Makey Makey https://www.youtube.com/watch?v=rfQqh7iCcOU