One solution for voice commands on Android based STB

Jovana Simić, Đorđe Glišić, Nikola Vranić, Marija Jovanović

***Abstract*—With the introduction of voice recognition and its support by various research groups, it became possible to add voice commands to different devices in use. We are seeing them on PCs, phones, and tablets. This paper presents a solution to order and control set-top box devices and TVs based on Android OS. It is a cloud-based solution supported by Google API for voice recognition.**

***Index Terms*—set-top box, android, voice commands, voice recognition, TV voice commands, Actions on Google, Android, Google Assistant, Dialogflow, Firebase.**

I. Introduction

Virtual assistance is becoming more popular with artificial intelligence and machine learning advances. Particularly with voice recognition available on PC and even mobile devices, with capabilities to support more languages, not just English. Popular operating systems come with support for virtual assistance and voice recognition as one service. There is Cortana on Windows operating system. In iOS, there is Siri. For Android devices and Google products, there is Google Assistant. Additionally, Samsung has Bixby, and Amazon has Alexa.

All mentioned virtual assistance is based on cloud architecture that employs the internet. Some solutions are standalone and work reasonably well in a narrow field, but cloud-based solutions give better results for general-purpose tasks.

This paper will investigate the possibilities of adding virtual assistance to set-top box devices to speed operations with DTV sets and improve user experience. We have selected an Android-based device with a remote controller that supports voice recording [1]. Early work on the subject was done with [1] and [2]. More similar work was done on narrow areas for content playback on set-top box devices by Petrovic et al. [3] and Visekruna et al. [4], and Lazic et al. [5].

II. Section Title (e.g., The Method)

Google Assistant is Google's virtual assistant available on mobile phones, smart home devices, TVs, cars, etc. It has artificial intelligence and enables two-way communication in voice communication and text messages with the user. It can browse the Internet, set alarms and events, access device settings, display user account information, etc. It also supports a lot of functionality for smart homes. For example, the user can control lighting, temperature, TV, and other items in the house. To prevent the service from being constantly active, the recognition of the keywords "OK, Google" and "Hey, Google" was introduced, after which the service became active. After activating Google Assistant, the voice command recording begins, and it collects an audio message which is further processed. The resulting response is reproduced depending on the device on which Google Assistant is used, e.g., in the case of Google Home devices, the answer is in the form of audio sound, while when using a smartphone or TV, a text record is also obtained [6]. If you want to run Google Assistant on an Android TV, you need a microphone to record your voice, integrated into the remote control [6].

Actions on Google (AoG) is a platform that allows you to expand your personal Google Assistant by adding your services called Actions. To enable device management, Action has been implemented. Suppose you want to access the Action described in this paper. In that case, you need to tell Google Assistant: "Talk to True TV" or "Ask True TV", after which he asks the AoG platform to run the application, AoG sends a request to the web service and receives a response forwards to Google Assistant, which displays the answer to the user. Furthermore, Google Assistant forwards the user input directly to the Action, and the Action responds directly to the Assistant.

When creating an Action, it is also possible to use the Dialogflow platform to simplify the understanding of user input. The Dialogflow agent translates user input during a conversation into structured data that applications and services can understand. As the call center, they are trained to handle expected conversation scenarios. The platform itself provides the ability to reply with static responses. If you want to respond with dynamic answers and additional logic, it is possible to implement a web service. In this case, Dialogflow is a proxy between Actions on Google and the web service, as shown in the figure. Instead of sending the request directly to the web service, AoG sends it to Dialogflow. Also, the web service sends a response to Dialogflow, which forwards it to AoG.

The answer must be returned to the user within 10 seconds. Otherwise, the request will expire. Also, the response must be less than or equal to 64 kilobytes. Dialogflow processes the user's input, and it sends an HTTP POST request to the web

Jovana Simić – RT-RK Institute for Computer Based Systems, Novi Sad, Srbija, (e-mail: Jovana.Simic@rt-tk.com)
Đorđe Glišić – RT-RK Institute for Computer Based Systems, Novi Sad, Srbija, (e-mail: Djordje.Glisic@rt-tk.com)
Marija Jovanović – RT-RK Institute for Computer Based Systems, Novi Sad, Srbija, (e-mail: Marija Jovanović@rt-tk.com)
Nikola Vranić – RT-RK Institute for Computer Based Systems, Novi Sad, Srbija, (e-mail: Nikola.Vranic@rt-tk.com)

 ISBN 978-86-7466-930-3

service, which is set as a function on the Firebase platform. The Cloud Functions for Firebase service automatically starts the function when an HTTP request arrives.

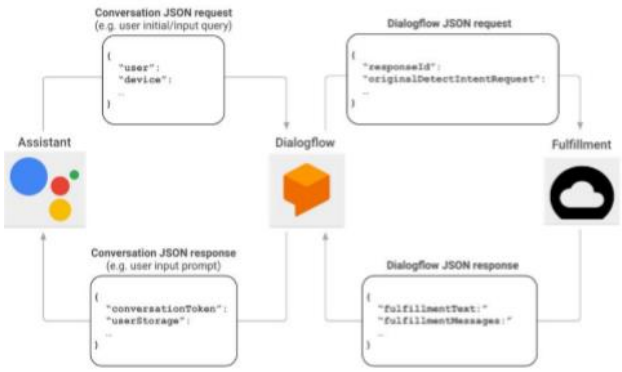

Fig. 1. The diagram shows the path that requests go from Assistant through the Dialogflow till the Fulfillment and response traveling back to the Assistant.

The Actions on Google, Dialogflow, and Firebase platforms are well integrated, as seen in Fig. 2. The figure shows a simple interaction between the end-user, the assistant, Dialogflow, and the web service built in the cloud, like Functions for Firebase and the Firebase Real-time Database.

The user enters his request as a voice command and starts the assistant himself. The assistant converts the voice command into text and starts the Dialogflow searching for the initial action (Welcome intent). Dialogflow uses machine learning to map text to intent and to isolate recognized entities (such as TV channel, TV channel number, time, etc.). Dialogflow triggers a web service, implemented as Cloud Functions for Firebase, sending it a JSON request that contains all the necessary information from the text. The web service processes the JSON request and implements logic (checks the Firebase Real-time Database) to respond to the assistant (or Dialogflow). Device Assistant converts the answer to speech and display (for devices that do not have a screen).

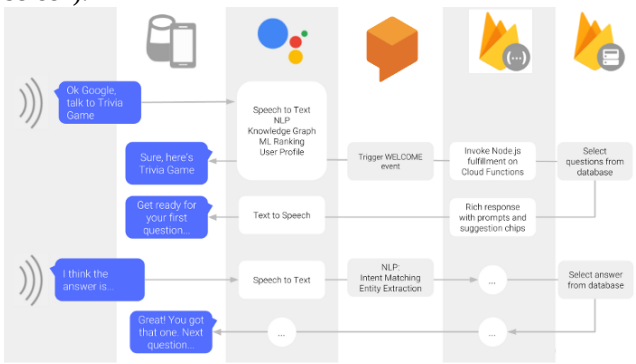

Fig. 2. Voice command workflow from Actions on Google, Dialogflow to Firebase fulfillment web service and back.

III. Implementation

This section describes one way to train and integrate Google Assistant. The Dialogflow agent needs to be trained first. The intent has been created for each supported command, in which it is necessary to enter a large number of phrases that the agent will understand, as shown in Fig. 3.

The pairing of the Action user and the STB device user was solved using a Google account, precisely his email address. To be able to manage the STB during the first launch of the Action, it is first necessary to link the account with the Action. The AoG platform provides the Google Sign-In option, the easiest way to connect and create accounts with the Action. The Action may request access to the user's Google profile, including the user's name, email address, and profile picture. Of course, you need to ask the user if he agrees to access his Google profile.

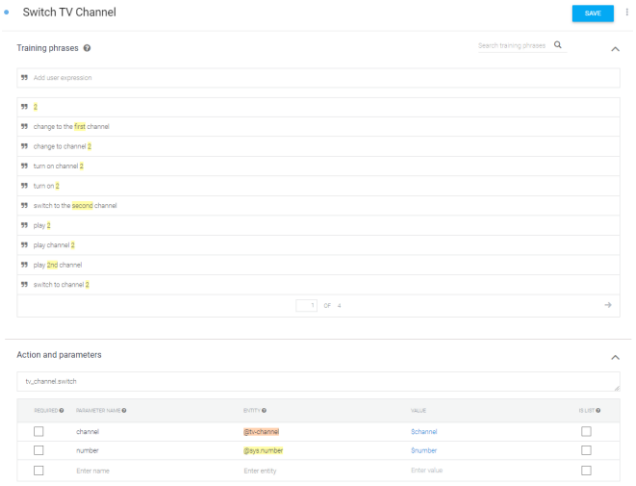

Fig. 3. Training phrases for specific action in Dialog flow and the required parameters defined.

After recognizing the command by the Dialogflow platform, it is necessary to send a signal to the TV application, which should execute the given command. Therefore, for each intent in the agent, an intent handler has been created in the web service. The operator first checks whether the user has linked his account with the Action. If this is not done, the user is first asked to link his account, and the user receives a list of suggested cards with commands for linking the account. Otherwise, the triggered command is entered in the Firebase Realtime Database database in a node whose email field value equals the user's email address accessing the Action. If such a node does not exist, the user with that email address is not logged in to any TV application. If there is, wait a while for the TV application to process the command and return the answer to be displayed to the user.

As already mentioned, the pairing of users of the Action and users of STB devices was done through a Google account. Therefore, the *AccountHandler* class is implemented in the TV application, which detects adding and removing accounts from the application. The class diagram of the *AccountHandler* class is shown in Fig. 4. Each account should have its node in the Firebase Realtime Database. Each account is associated with an object of the type *VoiceHandler* given in Fig. 5, which is used to detect database changes and execute commands.

 ISBN 978-86-7466-930-3

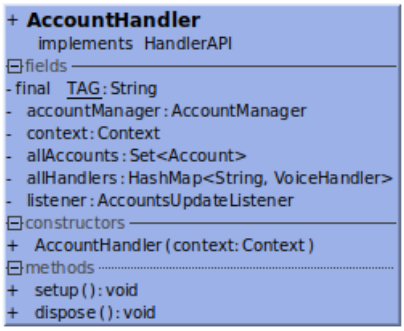

Fig. 4. Account handle class diagram.

When a new account is added to the TV application, it is necessary to add a new node to the database for that account and set the application to listen for changes on that node, for which the *VoiceCommand* class is used. In the class constructor, a reference to the root node of the database is retrieved, and in the *subscribe()* method, a listener is registered to the node in the database for which changes need to be detected.

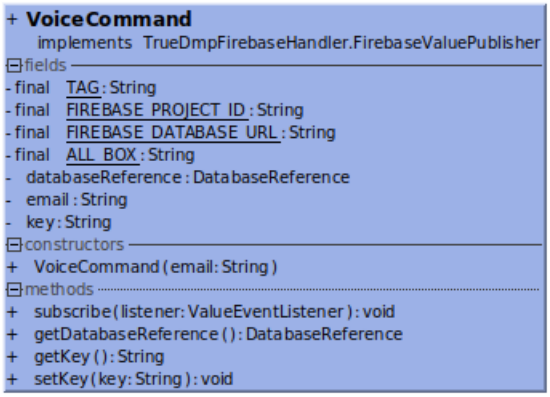

Fig. 5. Voice command class diagram.

IV. Results

A vital result of the presented work is that the voice commands apply to set-top-box devices with acceptable latency. Additionally, the phrase database could be upgraded without changing the actual code on any STB device.

Testing was performed via the Google Assistant application on a mobile device. The response time of the Action from the user's input to the return of the response to the user was tested. The tests were performed under normal conditions, under certain noises, using commands in English.

The table gives a comparative overview of the response time for commands implemented in this system. Response time averages 1 to 2 seconds, depending on the command. As shown in Table 1, the most time-consuming commands are for displaying information about events. These timings are comparable with the times it takes for users to do those actions using RCU. Key benefits could be seen in more complex scenarios, like adding services to favorite lists or scheduling PVR recordings based on the TV show's name for a specific period. Those cloud-based services will outperform manual user interaction with the graphical user interface.

TABLE I
VOICE COMMANDS AND RESPONSE TIME

Command for set-top box	Response Time[s]
Change to the next channel	~1,174
Change to the previous channel	~1,172
Change to a channel with a specific name	~1,195
Change to a channel with a specific number	~1,222
Complete tone reduction	~1,156
Restore volume before complete mute	~1,190
Tone amplification	~1.075
Mute the tone	~1,199
Increase the tone to maximum	~1,036
Check if the STB is in sleep mode	~1,353
View information about the current event on the current channel	~1,535
View information about the current event on a specific channel	~1,599
View information about the next event on the current channel	~1,469
View information about the next event on a specific channel	~1,445
Setting reminders	~1,169

V. Conclusion

In this paper, we have implemented a proof of concept for the voice commands on STB. For more implementation details refer to works [5], [6], and [7]. As with any cloud-based service, it relies on the internet and its stability. Further work shall be done to benchmark latency depending on the internet connection speed, location, and type of connection (WiFi, cable, mobile). It makes that operators supporting this feature offer stable enough internet.

If a set of basic commands is appropriately selected to cover all actions that the user can execute, then a cloud solution can make a chain of commands based on the user's voice request. Dialog flow cloud platform could process arbitrary complex requests that could be transformed and fulfilled as a series of actions. This hides a real benefit, as requests like "make a list of my top five sports channels and record NBA finals on internal disk" could be executed.

Additionally, requests like "Find me a movie with Tom Hanks for Saturday evening" contains more than just a request for STB. They hold user preferences so that operators can train recommendation systems. It opens up a vast field of opportunities as well as privacy concerns.

It does not end with voice commands for STB. It can be a command supported by a third-party application for food delivery or a calendar application.

 ISBN 978-86-7466-930-3

REFERENCES

[1] A. B. Garayalde, MSc thesis, “Speech Control & Media Sharing for Media Centers”, ENST Bretagne France, 2007

[2] KH Lin, CH Lin, KH Chung, KS Lin , “A Compressive Sensing-based Speech Signal Processing System for Wearable Computing Device in IPTV Environment”, ICMT 2013

[3] D. Petrović, M. Zeković and N. Vranić, "One solution for extension of the system for recording multimedia content on Android based devices" 2017 25th Telecommunication Forum (TELFOR), Belgrade, 2017, pp. 1-4.

[4] U. Višekruna and M. Savić, "Integration of Google Assistant in Android Application for Voice Control of Media Playback," 2018 26th Telecommunications Forum (TELFOR), Belgrade, 2018, pp. 1-4.

[5] A. Lazić, M. Z. Bjelica, D. Nad and B. M. Todorović, "Google Assistant Integration in TV Application for Android OS," 2018 26th Telecommunications Forum (TELFOR), Belgrade, 2018, pp. 420-425.

[6] E. Nan: „Upravljanje pametnom kućom uz pomoć Google asistenta”, University of Novi Sad, Faculty of Technical Sciences, Novi Sad, 2017.

[7] J. Simic, “Realizacija glasovnih komandi za uredjaj baziran na operativnom sistemu Android”, UB ETF, Belgrade, Serbia, 2022

 ISBN 978-86-7466-930-3