2017

Assistant Suite

Dale Greene  
*Virginia Commonwealth University*

John Pham  
*Virginia Commonwealth University*

Ritvik Nag  
*Virginia Commonwealth University*

Follow this and additional works at: [https://scholarscompass.vcu.edu/capstone](https://scholarscompass.vcu.edu/capstone)

Part of the [Electrical and Computer Engineering Commons](https://scholarscompass.vcu.edu/capstone)

© The Author(s)

Downloaded from  
[https://scholarscompass.vcu.edu/capstone/191](https://scholarscompass.vcu.edu/capstone/191)

This Poster is brought to you for free and open access by the College of Engineering at VCU Scholars Compass. It has been accepted for inclusion in Capstone Design Expo Posters by an authorized administrator of VCU Scholars Compass. For more information, please contact libcompass@vcu.edu.
Background

Objective
Produce an application that successfully demonstrates software-to-hardware implementation with the following points in mind:
• Providing a convenience for the user by providing a hands-free application
• Utilizing voice-to-mechanical implementation across multiple platforms within one application
• Advancing research in the direction of hands-free technology

Hardware used:
• Amazon Echo Dot
• Raspberry Pi
• RC Car Chassis Kit

Software used:
• Amazon Web Services (AWS)
• Alexa Skills Kit (ASK)
• Alexa Voice Service (AVS)

Future Goals
• Improve latency of voice command to mechanical movement
• Integrate a larger number of devices & microcontrollers into the application for increased versatility
• Add additional types of hardware to demonstrate versatility
• Allow AWS to communicate with multiple microcontrollers at once
• Test application on voice service platforms other than Amazon
• Test application with microcontrollers other than Raspberry Pi

High-Level Implementation Overview

Phase One: User speaks into Echo
Alexa is always listening and running against the Alexa Voice Service (AVS), which is a service embedded within the Alexa Developer portal, as long as there is a good internet connection since it is cloud-based. Once the "wake" word is spoken, Alexa at that point records what is said until the expected speech timeout is reached; the recording is then stored in AVS as a multipart message, consisting of a JSON-formatted object and a binary audio captured by the microphone.

Phase Two: Echo communicates with AWS
Once a new recording is stored, AVS then interacts with Alexa Skills Kit (ASK) to listen out for utterances that are manually stored in the developer portal. The results of whether or not a match to an utterance was found is sent to a linked lambda function or to send recorded voice to AWS.

Phase Three: AWS communicates with Raspberry Pi
There is one Raspberry Pi microcontroller connected to each individual device. All of the microcontrollers continuously check for new emails that arrive in a dedicated email account. The email sent from the lambda function determines which microcontroller is contacted and what segments of code are run within that microcontroller. The code that resides on each microcontroller is what drives the mechanical movements of each device. The microcontroller sends voltages to various motors in order to make the device move as instructed. All in all, the user specifies which device to control, and that specific device is channeled and moves specifically as instructed as the end result.

Background

Objective
Produce an application that successfully demonstrates software-to-hardware implementation with the following points in mind:
• Providing a convenience for the user by providing a hands-free application
• Utilizing voice-to-mechanical implementation across multiple platforms within one application
• Advancing research in the direction of hands-free technology

Hardware used:
• Amazon Echo Dot
• Raspberry Pi
• RC Car Chassis Kit

Software used:
• Amazon Web Services (AWS)
• Alexa Skills Kit (ASK)
• Alexa Voice Service (AVS)

Future Goals
• Improve latency of voice command to mechanical movement
• Integrate a larger number of devices & microcontrollers into the application for increased versatility
• Add additional types of hardware to demonstrate versatility
• Allow AWS to communicate with multiple microcontrollers at once
• Test application on voice service platforms other than Amazon
• Test application with microcontrollers other than Raspberry Pi

High-Level Implementation Overview

Phase One: User speaks into Echo
Alexa is always listening and running against the Alexa Voice Service (AVS), which is a service embedded within the Alexa Developer portal, as long as there is a good internet connection since it is cloud-based. Once the "wake" word is spoken, Alexa at that point records what is said until the expected speech timeout is reached; the recording is then stored in AVS as a multipart message, consisting of a JSON-formatted object and a binary audio captured by the microphone.

Phase Two: Echo communicates with AWS
Once a new recording is stored, AVS then interacts with Alexa Skills Kit (ASK) to listen out for utterances that are manually stored in the developer portal. The results of whether or not a match to an utterance was found is sent to a linked lambda function or to send recorded voice to AWS.

Phase Three: AWS communicates with Raspberry Pi
There is one Raspberry Pi microcontroller connected to each individual device. All of the microcontrollers continuously check for new emails that arrive in a dedicated email account. The email sent from the lambda function determines which microcontroller is contacted and what segments of code are run within that microcontroller. The code that resides on each microcontroller is what drives the mechanical movements of each device. The microcontroller sends voltages to various motors in order to make the device move as instructed. All in all, the user specifies which device to control, and that specific device is channeled and moves specifically as instructed as the end result.