Smart Home Automation through Voice Interaction

Andrei Mădălin Matei
University "Alexandru Ioan-Cuza" Iași
andrei.matei@info.uaic.ro

Adrian Iftene
University "Alexandru Ioan-Cuza" Iași
adiftene@info.uaic.ro

ABSTRACT

"Smart home automation through voice interaction" is a custom skill for well-known voice assistant Amazon Alexa. With its help, the user should simplify the way he is interacting with home devices, for example, TV, air conditioner or even lights. The application itself is based on the clouding programming paradigm along with devices like Arduino and Raspberry Pi. The main idea of this research is to build a smart house system where devices can be controlled with voice commands. The presented research tends to automate day-by-day tasks which may be trivial and helps the user to increase his quality of life by letting the system doing the work for him.

Author Keywords

Amazon; IoT; Smart House; Arduino; Raspberry Pi.

ACM Classification Keywords

H.5.m. information interfaces and presentation (e.g., HCI): Miscellaneous.

General Terms

Human Factors; Design; Measurement.

INTRODUCTION

This work, having its foundations in one of the most well-developed IT domains in recent years: IoT ("Internet of Things") [4], [5], shows how integration of well-known systems such as Amazon Alexa, Raspberry Pi[1] and Arduino automates some things a normal person does in the house, including interaction with the TV, interacting with an electronic device in the example: *a lamp, plant irrigation*, but also information such as *temperature* and *humidity* in the room, everything through voice interaction [2]. The idea for this came because we desired to simplify some processes that we are doing regularly and try to automate them. Besides this, we always wanted to try a non-conventional technology stack and combine them in various ways (Microsoft Azure with Amazon Alexa, debated in a special section).

Initially, the project was designed as an efficient way to control the television with voice using Alexa[2]. By the time that we have worked on the application, we realized that the potential was huge, so we added new sensors and an Arduino.

In the last part of the development process a curiosity that may be impacted all the application course "*What if* we *can create and maintain the* ***back-end*** *on another platform than Azure*?".

The above-mentioned IT domain, IoT has seen massive growth in recent years due to the increase in the number of devices connected to the internet [6], [7], and [8]. Not to mention the investments made in this sector, as we can see below.

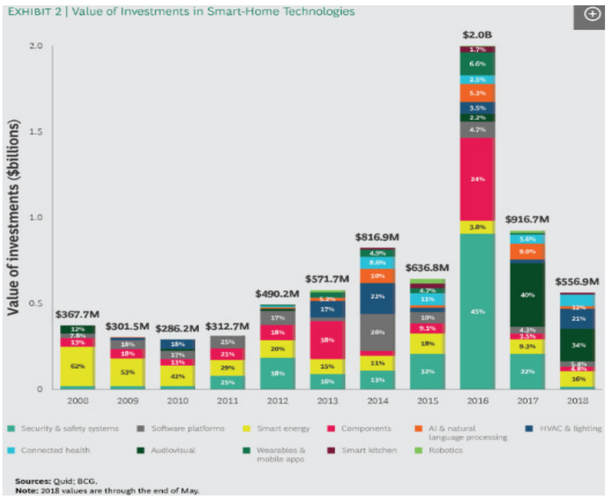

Fig 1. Investments made in "smart home" sector[3]

The evolution of the field also can be seen in Fig 2, where it is noticed that in 1992 already 1,000,000 devices used IoT technologies and in 2003 will reach half a billion. The number of devices has increased exponentially reaching 28.4 billion in 2013 and is projected to increase to 50 billion in 2020.

---

[1] https://www.instructables.com/id/DHT11-Raspberry-Pi/

[2] https://developer.amazon.com/docs/custom-skills/speech-synthesis-markup-language-ssml-reference.html

[3] https://d2p8144lghjt26.cloudfront.net/Growing-Smart_Home-Market_ex02_tcm-203601.png?mtime=20181008100744

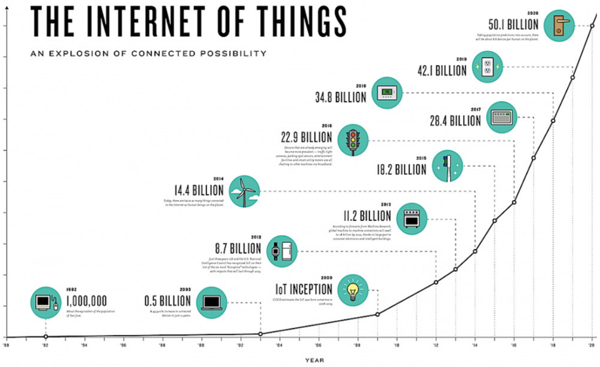

Fig 2. Internet of things expansion[4]

About Amazon Echo

Echo Dot is Amazon's response to this market, it uses Alexa's voice assistant, which through an interface made available by Amazon can create skills[5], that is, and one can learn the device to answer questions, say riddles, all of "*its memory is located in the cloud*". Besides this, major world companies, alongside Amazon, have begun developing a number of similar devices, such as Google Home, Apple Home Pod, along with their own voice assistants, Google Assistant and Siri.

About Raspberry Pi

Raspberry Pi is a product created by a charitable foundation in the UK. It resembles a lot of personal computers, with a small mention, is the size of a phone, making it ultra-portable and easy to install [2]. It is noteworthy that it has a reasonable computational power with a 1.2 GHz Quad-Core processor, 1 GB of RAM and a high-performance graphics accelerator, specs for the Raspberry Pi Model 3 B model. On the connectivity side, it has Wireless LAN, Ethernet, Bluetooth, 1 HDMI port, 4 USB ports, and Micro SD slot. The official operating system for this computer is Raspbian, a Linux distribution.

One thing to note about this device is the presence of 40 GPIO (General-purpose input/output) pins. These are mainly used to connect new sensors and devices, as demonstrated in the practical part of the license.

Since its launch in 2012, when Raspberry Pi Model 1 has appeared, so far, many varieties have emerged from the point of view of technical specifications. Since its launch in 2012, when Raspberry Pi Model 1 has appeared, so far, many varieties have emerged from the point of view of technical specifications.

About Arduino

Arduino can be regarded as the smaller brother of Raspberry Pi [3], because technically it is much more limited: 2 Kb SRAM and 16 MHz processor for the UNO model[6]. This is an open-source integrated circuit that reads and sends signals through its 14 pins. Arduino comes with many cost advantages but also the ease with which the programmer configures its work environment because it does not have a properly operating system, it communicates with the host computer via the USB port and the software available in the official documentation: Arduino IDE[7]. A disadvantage for Arduino would be the lack of a variety of programming languages, this being limited to C ++ and Java, which does not make it accessible to all programmers.

SIMILAR APPLICATIONS

The development potential is huge and varied, so there are applications in most areas, for example: medical, social and logistics. Applications fall into three broad categories, namely smart, industry-friendly cities, and those used by regular users: healthcare and intelligent homes. Intelligent systems that help citizens are implemented in the big cities, so the level of comfort increases. Some examples of such applications: intelligent tax payment systems, intelligent parking systems (which can find and manage free parking spaces in a car park), ticketing systems in the means of transport. The medical field also uses intensive technologies for: tracking medical equipment, securing salons in hospitals, tracking people with disabilities or diagnosing them at a distance. Below are some applications for improving daily living, used by ordinary users.

Ring Alarm

Ring is a company selling various security solutions for the interior and exterior of homes. They sell a wide range of sensors such as motion sensors, flood sensors, contact sensors and a wide variety of cameras, all to ensure residents' safety. In the Amazon skill store, there is also the Ring app[8], which by adding it to the user's account, can control the above mentioned smart devices.

Smart Life

Another popular app in the Amazon store, often used by people who want to interconnect smart home appliances, is Smart Life. The application involves buying smart devices from the manufacturer's website, such as plugs and switches. They connect to the wireless internet. in addition to the products on the site, other industry companies such as

[4] https://www.ncta.com/sites/default/files/styles/article_detail/public/2017-04/growth-of-internet-of-things-hero.jpg?itok=wgP_DGq

[5] https://developer.amazon.com/docs/ask-overviews/build-skills-with-the-alexa-skills-kit.htm

[6] https://classes.engineering.wustl.edu/ese205/core/index.php?title=Serial_Communication_between_Raspberry_Pi_%26_Arduino

[7] https://www.abr.com/what-is-rfid-how-does-rfid-work/

[8] https://shop.ring.com/collections/security-system

Koolertron and Oittm produce many devices that mate with Smart Life[9], including: intelligent cameras and light bulbs. The skill developed for this application has support for most devices, which are very easy to control by voice interaction.

Smart TV Remove

Smart TV Remote[10] works with the user's phone, turning it into a smart TV remote control. To use the skill, you must download the application with the same name in the Android store. The application has some limitations, including: it only works with Android devices, and the most important thing: the device must have an infrared sensor, and unfortunately the new devices are no longer equipped with such a sensor.

APPLICATION DEVELOPMENT

This chapter is dedicated to the entire architecture of the application along with its components and interaction. Here we chose to focus on the services we have been using, how to use the application, how to test rigorous motivation for the technically choices that we have made.

Some aspects covered here are:

- how to interact with the business side of the application using the two serverless solutions: Aws Lambda and Azure Functions;
- how to collect data for each user action;
- how to communicate with Raspberry Pi and Arduino;
- utility and need of sensors present in the project;
- how a user is notified of an action happening in his home;
- other sensitive points such as NFR (nonfunctional requirements).

Figure 3 presents an overview of the application architecture highlighting the key elements that have led to its realization.

Voice Service

As mentioned earlier, this part deals with the processing of natural language and its transformation into commands that the business part of the application understands. Initially, if a physical device has been purchased: for example, Echo Dot, the user is directed to the page where he can begin to create new skills for Alexa: https://developer.amazon.com/alexa/. After entering the name and language, the user has the option to select a predefined template with settings for different areas of interest: Music, Video, and a more simplified version for Smart Home. When selecting Custom, a skill is created without any predefined application facility.

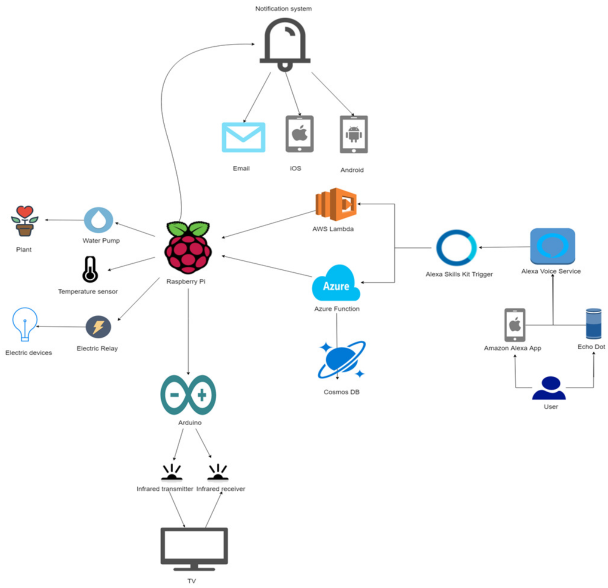

Fig 3. Application architecture

The application's name serves as a way to invoke it within the application, so the user starts the skill with one of the following invocations "*open smart home*", "*ask smart home"*.

Intents definition

Intents define the actions that the application in question can execute. Figure 4 shows a capture of the portal with the actions the application can execute.

Intents

+ Add Intent | Filter Intents

NAME	UTTERANCES	SLOTS	TYPE	ACTIONS
AMAZON.FallbackIntent	-	-	Built-In	Edit | Delete
AMAZON.CancelIntent	-	-	Required	Edit
AMAZON.HelpIntent	1	-	Required	Edit
AMAZON.StopIntent	2	-	Required	Edit
AMAZON.NavigateHomeIntent	-	-	Required	Edit
TurnOnIntent	1	1	Custom	Edit | Delete
TurnOffIntent	1	1	Custom	Edit | Delete
GetInformations	2	-	Custom	Edit | Delete
CustomCommand	1	1	Custom	Edit | Delete
WaterFlower	3	-	Custom	Edit | Delete

Fig 4. Screen capture with application intent in the https://developer.amazon.com/

9 https://www.amazon.com/Tuya-Inc-Smart-Life/dp/B01N1ZVI7M

10 https://play.google.com/store/apps/details?id=com.adi.remote.phone&hl=ro

As you can see, the application, by default, creates some of these actions that are handled in the next part of the application. These include the way to stop the application (AMAZON.Stopintent) that stops an action on the fly (AMAZON.Cancelintent) and the one to ask for help (AMAZON.Helpintent). Besides, we needed the following to build this application: *TurnOnintent* and *TurnOffintent*, used in the TV but also in the case of the Bulb, *CustomCommand*, used to execute the various start / stop commands for the TV, Get Information's (to request information on ambient temperature and humidity) and the last, *WaterFlower*, is the one to be used to operate the water pump.

Slots definition

In order to ease our work and not duplicate the code, as well as other intents, we have used the feature provided by the portal called *slot*[11]. By doing so, we have defined a number of devices that the actions receive as a part of it, so we managed to gain time and structure the code better. In Figure 5 are the two slots we have created.

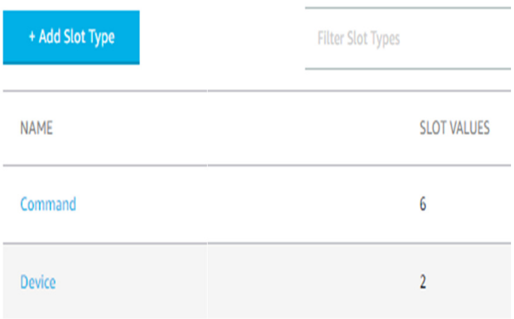

Fig 5. Screenshot with slots used

The two contain a list of devices, in the case of Device and commands in the case of Command. In order to reuse *TurnOnintent* and *TurnOffintent*, we chose to send them a Device that lists the following: TV and light, so we took full advantage of it and did not bring unnecessary complexity. The next slot contains a list of commands the user can give to the TV, in addition to those on / off, these are: volume up, volume down, channel up, channel down, mute and unmute. Within these slots, a dictionary for synonyms can also be defined to cover a wider range of phrases.

Utterances definition

These are the phrases that the user is using to utter an action, defined above. It should be noted that they are as diversified and accept as many variations as possible from the basic invocation, for example: "*Turn on the TV*", "*turn on the television*" the user does not have to memorize the phrases, and these comes naturally. It is noteworthy that depending on the programming language version, how these phrases should be said and how they need to be related to the application name can easily vary, so a parallel of this kind is debated in the following chapters.

Skill Service

This part of the research presents the logic of the application and how to interact with the voice service. As a programming language we used .NET Core due to the knowledge gained in the faculty, but the fact that it is a technology that is where Microsoft invests massively, so development is intense. Initially, when we started the work for the application, there is no support for .NET Core 2.0, so we opted for version 1.0. It was officially flagged by Amazon as being deprecated in April 2019, so we switched to the new version. As notable differences, strictly within this application, is how to invoke the application invocation followed by the command. One example is that: in the .NET Core 1.0 version, the application name should be repeated each time it was intended to execute a command: "*ask smart home turn on the light*", "*ask smart home turn on the TV*". By comparison, in the .NET Core 2.0 version, this has been solved so that now everything is reduced to opening the application, naming its name, and then just the name of the order. In our case, the scenario is the following: "*open smart home*", "*turn on the light*", and "*turn on the TV*".

Aws Lambda Configuration

In order to create such a function, the user needs an account on the dedicated aws page: https://aws.amazon.com/. Here is the name of the function, the programming language, and some permissions for it. Figure 6 is a screen capture with this functions at this time on the Aws Lambda portal, where the two, with the related version [1].

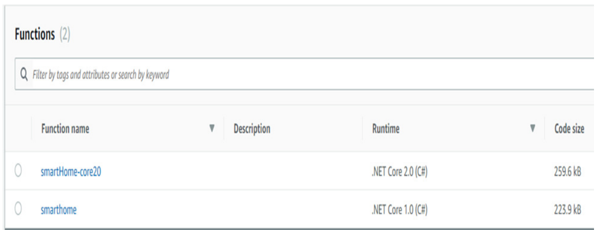

Fig 6. Screenshot from Aws Lambda portal

Voice Service integration

In order for the functions described above to communicate with the voice service, they need to know how to capture and treat requests differently according to some parameters: the

[11] https://developer.amazon.com/docs/custom-skills/create- intents -utterances-and-slots.html

type of request, the type of intent, and the type of slot. The answer given by the function is a combination of these.

In .Net Core, the Alexa.Net package, requests from Alexa can be two-fold: *LaunchRequest*, this is the application that the user encounters when the application is open and provides some information about what the application is capable of doing. Through *IntentRequest* it can differentiate requests, so the application know that it received a different request from the presentation, and therefore it has to treat it correctly. The *IntentRequest* object contains the intent attribute, so depending on its Name attribute, it can differentiate what type of intent the application receives. In the case of slots, things are similar. The intent object described above has as an attribute a dictionary (*name*, *value*), and through it it can access the value of an intent by its name.

Raspberry Pi and Arduino

Initially, when we started this application, the Raspberry Pi integrated board was not part of the architecture or future plans for it [3]. One problem we faced was to connect the Arduino card to the internet because it does not come with an attached network module. Another problem was that after connecting it to the internet, there was some difficulty in making requests and receiving HTTP responses because the library does not have much documentation or support for object-oriented programming. This problem has been resolved by Raspberry Pi. Through it, the issue of requests and answers has been solved in several ways: First, there is the option of connecting it to a network, either Wi-Fi or cable, via Ethernet. Secondly, the variety of languages is much higher compared to Arduino that only supports C. For this, we chose to use the Python programming language, version 3.7, because it is suitable for applications of this dimension, but also because of the Flask framework.

Another method of comparison between the two microcontrollers is also the speed with which someone can actually create an application / service through them. In this chapter, Arduino was much easier because it only connects with the USB cable from a personal computer and the dedicated application sends the data to it. In contrast, at Raspberry Pi, we need an image of a Raspbian operating system, and basically you need to take steps similar to installing any another operating system. From these two ideas, the actual time to install one of the two is considerably higher at Raspberry, given its technical specifications. The communication with it was made easy through the VNC Viewer application by entering its address and the account created on the machine.

Another method of communicating more efficiently when it does not want to interact with the plate display, but only sending bash commands to the command line, is also PuTTY, which connects through the SSH protocol to port 22.

Figure 7 represents a non-finite state of the system. As you can see below the water pump or the electric relay was not attached yet.

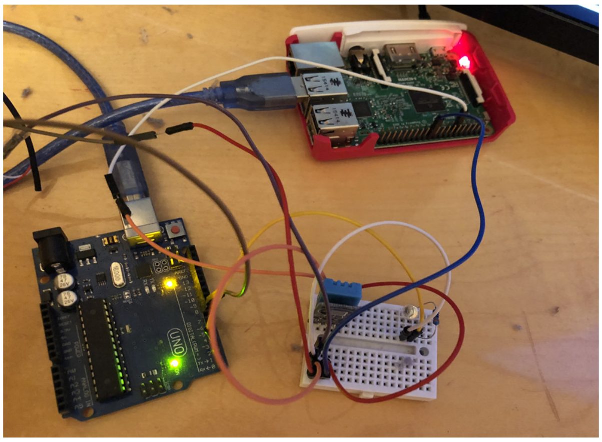

Fig 7. Non-finite state picture

CONCLUSIONS

The work presented with the name "Smart Home solution" keeps lifestyle easier by diminishing the interaction between the user and some components in his home. With this easy and friendly way of voice interaction, the user can control their electronic devices wherever they are and more, they can get details about real-time sensors in the home. This type of interaction mentioned above is due to the demands made on this segment by users who want simple ways to automate some processes. This work tends to express a culture rich in the types of technologies used so that the clouding programming paradigm dominates most of the application and the motivation for it is found in the dedicated chapter.

The development potential for the application presented is only the number of devices and sensors with which it can integrate. As a result of massive development in this segment and the emergence of new electrical devices and sensors, the application has a huge potential, given the type of architecture chosen for it. Some examples might be: connecting to several devices that support infrared transmission: air conditioning, projectors, integration with other household appliances: washing machines or clothes, smart vacuum cleaners or the central thermostat, or integration with some sensors that can be added to the house: humidity, carbon monoxide, gas or motion.

REFERENCES

1. Boricha, B. Serverless computing wars: AWS Lambdas vs Azure Functions. In Packtpub. Cloud & Networking News, (2018).
2. DeLisle, J. J. Top 8 Raspberry Pi alternatives and rivals. In Electronic Products. Board Level Products - Single Board Computer. (2017).
3. Di Justo, P. Raspberry Pi or Arduino Uno? One Simple Rule to Choose the Right Board. Makezine. (2015)
4. Hassan, Q. F. Internet of Things A to Z: Technologies and Applications. Wiley-IEEE Press. (2018)
5. Jindal, F., Jamar, R., Churi, P. Future and challenges of Internet of Things. International Journal of Computer Science & Information Technology (IJCSIT) vol. 10, no 2, April (2018)
6. Park, E., del Pobil, A. P., Kwon, J., Kwon, S. J. The Role of Internet of Things (IoT) in Smart Cities: Technology Roadmap-oriented Approaches. Sustainability 10(5):1388 (2018).
7. Porkodi, R., Bhuvaneswari, V. The Internet of Things (IoT) Applications and Communication Enabling Technology Standards: An Overview. International Conference on Intelligent Computing Applications (ICICA), pp. 324-329 (2014).
8. Wortmann, F., Flüchter, K. Internet of Things. Bus Inf Syst Eng vol. 57, no. 3, pp. 221–224 (2015).