Max, the Emergency Assistant based on Amazon Alexa

Alvaro Gomez Hernandez
Computer Science Master's Student
Illinois Institute of Technology
alvarogoher@gmail.com

ABSTRACT

In this paper it is described MAX, the emergency assistant. MAX is based on Amazon Alexa (the Amazon's voice control system), Amazon Web Services (cloud services) and Twilio (a platform to provide calling and messaging functionality to applications). It has been designed and implemented by Alvaro Gomez, Computer Science Master's student at the Illinois Institute of Technology in the moment of the project development. The mentorization and funding of the RTC Labs Director, Carol Davids has been indispensable to carry out the project. This project tries to resolve an in-home emergency problem: Often, when users need to call 911 when being in an emergency, they do not have immediate access to a phone or even to the signal to place the call. Those precious seconds, that can mean the difference between life or death can be saved by the use of MAX.
MAX is capable of placing standard phone calls, SIP calls and sending SMS. When the user is having an emergency a call to 911 (with valuable information about the user such as the user's name and address) can be automatically placed. MAX is invoked with voice commands such as "Alexa, open MAX" or "Alexa, tell MAX to make a call". When people make 911 calls they have to answer several questions before requesting help such as "what's your name", "where are you", "do you have allergies", etc. The answers to these questions are known before the emergency takes place and here is where MAX has found its market gap. MAX has been developed to help people, it is very useful for people with limited mobility, aged people, blind people or children among others. Although the basic functionality has been developed and MAX works, some future improvements need to be done in order to finish its development and get the whole potential power of MAX.

Author Keywords

IIT, RTC Labs, Amazon, Amazon Web Services, Amazon Lambda function, Alexa, Alexa Skill, MAX, Emergency Assistance, Emergency Calls, Automated Calls, Automated SMS, SIP calls, VoIP, 911, NG911, Node.js, Twilio, TwiML, Twilio REST API.

INTRODUCTION

This project was born in the Real-Time Communications Laboratories (RTC Labs) at the Illinois Institute of Technology (IIT) under the supervision of the RTC Labs' Director, Professor Carol Davids. At that moment, researchers were carrying out a project for making possible a better Next Generation 911 (NG911) for indoor locations.

This project tries to resolve an in-home emergency problem. Often, when users need to call 911 when being in an emergency, they do not have immediate access to a phone or even to the signal to place the call. Those precious seconds, that can mean the difference between life or death, are limited. Now, there is a smart emergency assistant capable of forwarding your need for assistance as well as your relevant information and do it almost instantly with a simple voice command. This is MAX.

MAX, based on Amazon Alexa, is a potential permanent listener of the users (at their homes) and whenever they need assistance, it can ask for help. In other words, MAX is an Alexa Skill with specific capabilities to interact with the users, collect their personal information and place an automated communication with a valuable message, both voice, and text, to any endpoint specified by the user.

One of the things NENA, the 911 association, tell us if a user places a call to 911 is: "The first thing 9-1-1 needs to know is location and type of help needed". If MAX knows that and other relevant information before the emergency would take place, users can save some valuable minutes that could mean the difference between life and death. But, What Is an Alexa Skill? "Alexa is Amazon's voice service and the brain behind tens of millions of devices like the Echo family of devices including Echo Show and Echo Spot. Alexa provides capabilities, or skills, that enable customers to create a more personalized experience. There are now tens of thousands of skills from companies like Starbucks, Uber, and Capital One as well as other innovative designers and developers."

When an Emergency would occur, then the user would invoke the skill which triggers an automated communication (with the name and complete address of the user in the body of the message) to a specific endpoint. This communication may be one of the three following ones, but it also may be the three of them at the same time:

- Standard call/s to 911.
- Standard call/s to a user/'s folk.
- SIP call to a specific domain.
- SMS to the user/s folk/s.

Basically, this would mean that the whole user's neighborhood may be able to know almost instantly that this user is having an emergency what sort of emergency is. The project is mainly formed by 3 components: The Voice User Interface developed with Alexa Skills Kit, the back-end of the Skill developed with AWS and the calling function developed with Twilio.

1.1. Technologies used in the project

The following list shows what technologies has been used to design and implement the project. Some of them are related with Amazon Web Services (AWS):

- **Amazon Alexa**. Alexa (named after the ancient library of Alexandria) is Amazon's voice-control system.
- **Amazon Echo Dot**. Echo Dot is a hands-free, voice-controlled device with a small built-in speaker which connects to the Alexa Voice Service to provide services instantly.
- **Alexa Skills Kit**. The Alexa Skills Kit is a software development kit (SDK) that enables a developer to build skills, also called conversational applications, on the Amazon Alexa artificial intelligence assistant.
- **AWS Lambda Functions**. AWS Lambda is a serverless compute service that runs user's code in response to events and automatically manages the underlying compute resources for the user. The user can use AWS Lambda to extend other AWS services with custom logic, or create user's own back-end services that operate at AWS scale, performance, and security.
- **AWS DynamoDB**. Amazon DynamoDB is a fully managed proprietary NoSQL database service that supports key-value and document data structures and is offered by Amazon.com as part of the Amazon Web Services portfolio.
- **AWS IAM**. AWS Identity and Access Management (IAM) enables the user to manage access to AWS services and resources securely. Using IAM, the user can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources.
- **AWS CloudWatch Logs**. Amazon CloudWatch Logs provides monitoring, storage, and access to user's log files from Amazon Elastic Compute Cloud (Amazon EC2) instances, AWS CloudTrail, Route 53, and other sources. User can then retrieve the associated log data from CloudWatch Logs.
- **Node.js**. Node.js is a free, open source server environment that runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.) using JavaScript on the server.
- **Apex**. Apex lets the user to build, deploy, and manage AWS Lambda functions with ease. With Apex the user can use languages that are not natively supported by AWS Lambda, such as Golang, through the use of a Node.js shim injected into the build. A variety of workflow related tooling is provided for testing functions, rolling back deploys, viewing metrics, tailing logs, hooking into the build system and more.
- **JSON**. JavaScript Object Notation (JSON) is a syntax for storing and exchanging data written with JavaScript object notation.
- **Twilio**. Twilio is a developer platform for communications. Software teams use Twilio API to add capabilities like voice, video, and messaging to their applications. This enables businesses to provide the right communications experience for their customers. Behind Twilio API is a Super Network, a software layer that connects and optimizes communications networks around the world.
- **Twilio TwiML**. TwiML (the Twilio Markup Language) is a set of instructions the user can use to tell Twilio what to do when the user receives an incoming call, SMS, or fax.
- **Twilio REST APIs**. Twilio's APIs (Application Programming Interfaces) power its platform for communications. Behind these APIs is a software layer connecting and optimizing communications networks around the world to allow users to call and message anyone, globally. The user can use the REST APIs to programmatically add capabilities like voice, video, and messaging to your applications.
- **SIP**. The Session Initiation Protocol (SIP) is a signalling protocol used for initiating maintaining and terminating real-time sessions that include voice, video and messaging applications. SIP is used for signaling and controlling multimedia communication sessions in applications of Internet telephony for voice and video calls, in private IP telephone systems, in instant messaging over Internet Protocol (IP) networks as well as mobile phone calling over LTE (VoLTE).

Figure 1. Logos of the Technologies used in the project

In the Figure 1, the logos inside the box, from left to right: Amazon Alexa, Amazon Web Services, Node.js, Twilio. In the Figure 1, the logos outside the box, from left to right: AWS Lambda, AWS IAM, AWS DynamoDB, JSON, Apex, SIP.

1.2. Cases of Use

This following list shows the possible cases of use where MAX can play an important role in helping users:

- **Sanitary Emergency**. The user needs health-personel to be sent.
- **Firemen Department Emergency**. The user needs firemen to be sent.
- **Policemen Department Emergency**. The user needs policemen to be sent.
- **Intrussion Emergency**. The user have reasons to think some intruder is at the house and needs the police to be sent by invoking the skill with a secret word.
- **All in Emergency**. The user needs health-personel, firemen and policemen to be sent.
- **Send a broadcast message to anyone**. The user needs to send a broadcast message to some end-users.

2. SYSTEM DESCRIPTION

2.1. System Operation

The Figure 2 shows the System Data-flow. The main interactions (from 1 to 4 labels in the Figure 2) are:

1. The interaction between the user and the Amazon Echo Dot. This is interaction is based on voice commands in both directions.
2. The interaction between Amazon Alexa and AWS Lambda is based on JSON input / output files.
3. The interaction between AWS Lambda and Twilio is based on HTTP POSTs.
4. The interaction between Twilio and the endpoint-user/s is based on standard phone calls, SIP calls, SMS or the three of them concurrently.

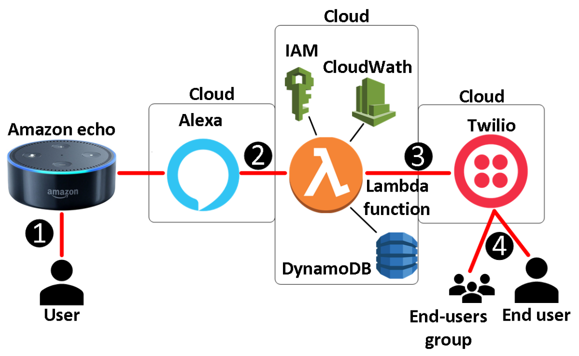

Figure 2. System Data-flow

2.2. Front-End: Voice User Interface

The Voice User Interface (VUI) defines the flow of all the possible interactions and paths the conversation can have between the Skill and the user. In all the figures related to the VUI (Figure 3, Figure 4, Figure 5 and Figure 6) the smooth boxes represent invocable commands by the user, the green boxes represent decisions MAX has to make, the boxes with straight corners and visible margins and black-color letters represent voice messages prompted to the user, the boxes with straight corners and visible margins and red-color letters represent actions taken by MAX. The VUI is created using the ASK. There are some concepts that need to be understood when creating a VUI in the ASK:

- **Name of the Skill**. Name to find the skill in the Amazon database.
  · **MAX's name**: Max (e**M**ergency **A**ssistant with ale**X**a).
- **Invocation name**. Word to invocate the skill.
  · **MAX's Invocation name**: ”max” (short, easy to remember, friendly).
- **Intents**. Represent actions that users can do with your skill. In other words, they are the core functionality of the skill. The Intents cane be both developed and built-in (the ones with the word ”AMAZON” in their names are built-in Intents). For this and further sections this ”X → Y” means ”X invokes Y” and this ”→ Y” means ”when Y is invoked”. This is only used when explaining Utterances and Slots.
  · **MAX's Intents**:
  → **KnownUserIntent**. It verifies if the user which is invoking MAX is a new user or it is already in the database. If the user is known, it prompts the options that can be selected by the user. Otherwise, it prompts the need for the user to set up in order to continue.
  → **SetUpIntent**. It prompts the first step of the setup process which is saving the user's name.
  → **NameIntent**. It saves the name of the user and then it prompts the second step of the setup process which is saving the user's address.
  → **DeviceAddressIntent**. It asks for permission to access the user's address saved at the Amazon account linked to the Alexa device which is being used. It also saves the address in the database if the user gave permission.
  → **CallIntent**. CallIntent. It places a call only if the database has the user's information stored in the database. Otherwise, it prompts the need for the user to set up in order to continue.
  → **DeleteIntent**. It erases the user's information from the database.
  → **AMAZON.FallbackIntent**. This Intent (available in English locales only) is triggered when the user's spoken input does not match any of the other intents in the skill. AMAZON.FallbackIntent is matched to an automatically-generated out-of-domain model. The AMAZON.FallbackIntent handler can give the user more details on what your skill does and how they can interact with it.
  → **AMAZON.CancelIntent**. This Intent is invoked to cancel a transaction, but remain within the skill
  → **AMAZON.HelpIntent**. This Intent is invoked to provide help about how to use the skill.
  → **AMAZON.StopIntent**. This Intent is invoked to stop the current skill
- **Utterances**. Specify the words and phrases users can say to invoke the intents. The developer maps these utterances to the intents. This mapping forms the interaction model for the skill.

· **MAX's Utterances**. These are the Utterances of the skill related to the Intents they invoke:
"{answer} I am not {name}" → KnownUserIntent
"{answer} I am {name}" → KnownUserIntent
"{answer}" → KnownUserIntent

"I would like {set_up_command}" → SetUpIntent
"{set_up_command} please" → SetUpIntent
"please {set_up_command}" → SetUpIntent
"{answer} {set_up_command}" → SetUpIntent
"{set_up_command}" → SetUpIntent

"{name} is my name" → NameIntent
"I am called {name}" → NameIntent
"I am {name}" → NameIntent
"My name is {name}" → NameIntent
"{name}" → NameIntent

"find my {address}" → DeviceAddressIntent
"{address}" → DeviceAddressIntent

"{answer} {call_command}" → CallIntent
"{call_command}" → CallIntent

"delete my persona information" → DeleteIntent
"I want to delete" → DeleteIntent
"I'd like to delete" → DeleteIntent
"delete please" → DeleteIntent
"delete" → DeleteIntent

- **Slots**. Words or phrases that represent variable information in the Utterances. I.e. When defining the Utterances the developer can use Slots to represent a set of words, for example, the slot name can be any person name defined by the developer in the Slot section.
·**MAX's Slots**:
{**answer**}:
·I am
·confirm
·sure
·yup
·no = don't = nope
·yes
{**call_command**}:
·calling
·just call
·call for me please
·I'd like to make a call
·I would like to make a call
·please call
·call please
·make a call
·call
{**set_up_command**}:
·set up
{**address**}:
·address
{**name**}: 5164 international person names (dataset from: `http://www.quietaffiliate.com/free-first-name-and-last-name-databases-csv-and-sql`)

- **Endpoint**. The identifier of the cloud-based service that accepts these intents as structured requests and then acts upon them. This service must be accessible over the Internet. The Endpoint will receive POST requests when a user interacts with the Alexa Skill. The request body contains parameters that the service can use to perform logic and generate a JSON-formatted response. It is possible to host an own HTTPS web service endpoint as long as the service meets the requirements. But the most common option by developers is the use an Amazon Lambda Function as an Endpoint using the Amazon Resource Names (ARNs). Amazon Resource Names (ARNs) uniquely identify AWS resources.
· **MAX's Endpoint**: Amazon Web Services Lambda Function ARN
→ arn:aws:lambda:us-east-1:XXX:function:XXX

To invoke MAX the user has two options. On the one hand, the first one is the longest one and it made to set up the information and to have an conversation-feel interaction with Alexa. This first option is invoke by saying "Alexa, open MAX". If there is data stored in the database, MAX will ask the user if is a known user. If the user is known, Alexa will welcome the user and will prompt the options that the user can choose (setup, make a call, delete). If the user is not known or there is not data in the database, Alexa will ask to setup. On the other hand, the second option is the shortest one and it is made to place a call when the user has an emergency and the user has already made the set up process. The Figure 3 shows this behavior, i.e. the first interaction between MAX and the user and all the possible sub-paths. It is important to highlight that although the Stop command is shown in the diagrams of Figures 3, 4 5 and 6 Alexa will never prompt this option, but it still is accessible.

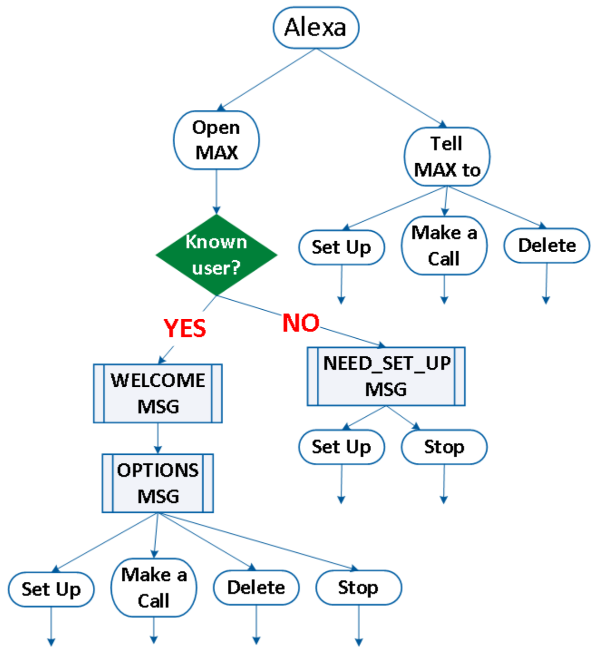

Figure 3. VUI - First Interaction

When the setup command is said, it triggers a path in the VUI which first asks for a name, then it asks for permission to find out the address of the user and it finally prompts the options that the user can choose (setup, make a call, delete). Figure 4 shows this behavior, i.e. the path when the user invokes the SetUPIntent and all the possible sub-paths.

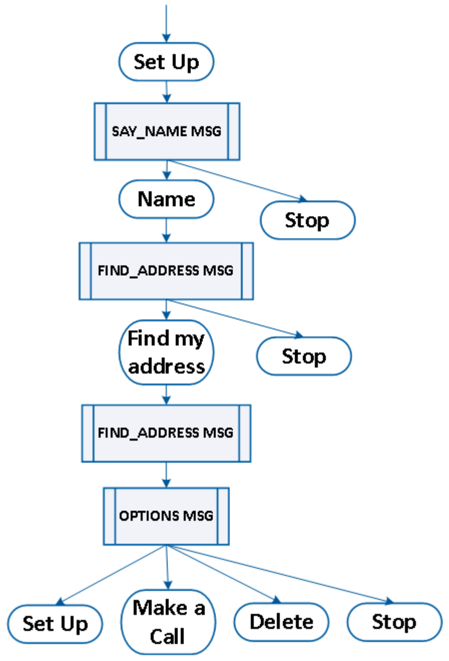

Figure 4. VUI - SetUpIntent

When the call command is said, it triggers a path in the VUI which it first checks if the user has data in the database. Then, it will make a call if there is a name and an address stored. Otherwise, Alexa will ask to setup. Figure 5 shows this behavior i.e. the path when the user invokes the CallIntent and all the possible sub-paths.

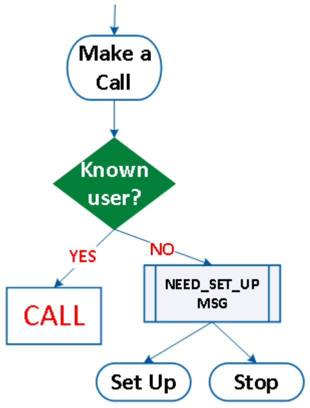

Figure 5. VUI - CallIntent

When the stop command is said (labeled as 1 in Figure 6), it triggers Alexa to prompt a goodbye message and then it exits the application. When the delete command is said (labeled as 2 in Figure 6), it triggers Alexa to prompt a delete message and then Alexa will ask to setup.

The Figure 6 shows this behavior, i.e. shows the path when the user invokes the StopIntent (1-label), the path when the user invokes the DeleteIntent (2-label) and all the possible sub-paths.

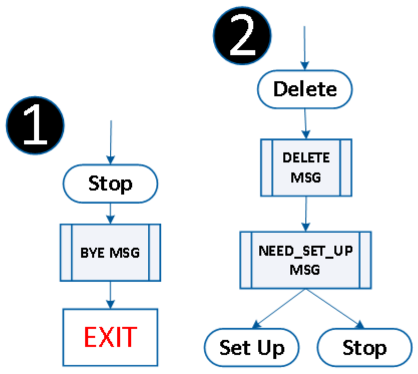

Figure 6. VUI - StopIntent and DeleteIntent

2.3. Back-end: Triggers, Logic, Data Managment

The back-end is allocated in the Amazon Cloud. It is mainly conformed by three components. The Figure 7 shows the three of them: Amazon Alexa AI (labeled as 1), the Amazon Lambda function (labeled as 2) and the resources that the Lambda function has access to like AWS DynamoDB or AWS CloudWatch (labeled as 3).

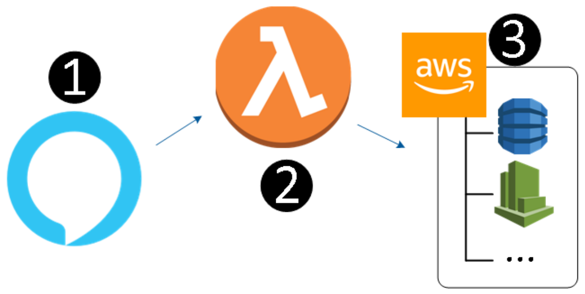

Figure 7. Back-end flow

When working in the dashboard of AWS, it is needed to select the trigger of the Amazon Lambda function. This is done by selecting "Alexa Skill Kit" from the options list. Then, it is needed to configure the trigger. It is needed to introduce the skill ID in the AWS-trigger dashboard and also the ARN of the Amazon Lambda function in the ASK skill dashboard. Finally, it is needed to introduce the ARN of the function in the ASK dashboard in the specific skill.

The Amazon Lambda function allocates the main logic of MAX. But, what is what Is AWS Lambda? "AWS Lambda is a computing service that lets to run code without provisioning or managing servers. AWS Lambda executes the code only when needed and scales automatically, from a few requests per day to thousands per second. The user pays only for the compute time consumption - there is no charge when the code is not running."

"AWS Lambda runs the code on a high-availability computing infrastructure and performs all of the administration of the computing resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging. The languages that AWS Lambda supports are Node.js, Java, C#, Go and Python".

The code of MAX is written in Node.js. AWS allows writing the code of Lambda functions in its own online editor, however, if the project requires external Node.js package it is a better practice to code in our local machine. For the MAX project, it has been used a technology called APEX. It allows using external Node.js packages to code in our local machine and then upload a .zip package with all the needed resources.In the Node.js module system, each file is treated as a separate module. The structure of the MAX's Amazon Lambda function follows some guidelines. These are listed below:

- Loading of the external modules (alexa-sdk, twilio, etc.).
- Handlers declaration and implementation. The handlers are the entry point of the Amazon Lambda function. In the MAX project here is the code called by the Intents of the Amazon Alexa skill.
- Core logic declaration and implementation. I.e. supporting functions for the handlers. In the MAX project here is the code to place a call or to send SMS among others.
- Initiation of Alexa with the handlers created.
- Execution of Alexa.

For further information about the Lambda function coding visit "AWS Lambda. Best Practices for Working with AWS Lambda Functions". The AWS resources accessible from the Lambda function include the following ones: AWS CloudFormation, AWS IoT, AWS Key Management Service, AWS Lambda, AWS XRay, Amazon CloudWatch, Amazon CloudWatch Events, Amazon CloudWatch Logs, Amazon Cognito Identity, Amazon Cognito Sync, Amazon DynamoDB, Amazon DynamoDB Accelerator (DAX), Amazon EC2, Amazon Kinesis, Amazon Resource Group Tagging API, Amazon S3, Amazon SNS, Amazon SQS, Application Auto Scaling, Data Pipeline, and Identity And Access Management. In the case of the MAX project is important to highlight the use of the DynamoDB. Amazon DynamoDB is a fully managed non-relational database service that provides fast and predictable performance with seamless scalability. MAX makes use of it to store the session attributes. The following table shows a record of a user.

DynamoDB item	User ID	mapAttr
Type	string	string
Number of elements	1	9
Elements	ID	address, address items

The ID of the user is an alpha-numeric string that uniquely identifies the user in Amazon. Whereas the address is a unique string formed by items on the Amazon account address section. The address items are these items stored singularly.

The decision of storing the data twice is because the address inserted by the user in the Amazon account may have blank spaces and Twilio does not support to receive blank spaces in the items which are sent in the HTTP POST, so the singularly-stored address values are filtered by a function (blanckReplacement function) that replace the blank spaces to underscore sign: "_". For example, blanckReplacement(750 North Rush Street) would return 750_North_Rush_Street.

2.4. Calling and messaging functionality: Twilio

The functionality to communicate with the endpoint-users is carried out with Twilio. Creating a premium account in Twilio and renting a phone number are the first steps. After some documentation process to understand how the Twilio REST API works, it is needed to include the code to place calls and send SMS as well as to create the TwiML code that will support the application. The rented phone number supports voice calls, fax sending, SMS sending and MMS sending.

In order to be able to use the capabilities of the Twilio account in the Amazon, Lambda function it is needed to include the Twilio user token and the Twilio user ID. Then, in the code, it is needed to specify the calling phone number - i.e. the rented number in Twilio - and the endpoint-number/s as well as the HTTP POST message that Twilio REST API will manage. In the body is sent the specific name and address of the user. The TwiML BIN includes the message that will be played to the endpoint-user when he/she answer the call. In this message are inserted both the name and the address (passed in the HTTP POST) making this message personalized. For example, if a user called "Will" would invoke MAX, then MAX would place a call with the following message:

"*Hello, this is an automated call to report an emergency. This device belongs to **Will** and this person lives in **Line1**, **Line2**, **Line3**, **District**, **City**, **Country**, Postal Code: **Postal**. Please, send immediately an emergency unit, the life of this person might be in danger!*"

Being Name, Line1, Line2, Line3, District, City, Country, Postal parameters passed in the HTTP POST sent by the Amazon Lambda Function to the TwiML REST API. And being the rest of the message stored at the Twilio cloud in form of a Twilio BIN. The messaging functionality works very similar to the calling functionality, but there is not need of a TwiML BIN. The body of the SMS is sent directly in the HTTP POST. The following table shows the capabilities of a unique Twilio phone number.

Item	Destination	Actions / Second
SMS-capability	US	1
Voice-calling-capability	US	1

This means that with a unique number it is possible to send a broadcast call / message with one unique phone number with a rate of 1 per second. The price for renting a phone number is $1 per month. The price to send a SMS is $0.0075. The price for make a phone call is $0.0130 per minute. For further information, visit the Twilio pricing section and consult the rates for your country.

2.5. SIP

MAX is also capable to place SIP calls. This allows to place calls to endpoints where there is not signal to receive standard phone calls, but there is a WiFi connection where the endpoint has a SIP user agent correctly registered. The code to place a SIP call is very similar to the code to place standard phone calls. In this case, the same TwiML code and TwiML BIN is used than the standard phone call. Twilio is able to place one SIP call per second. The RTC labs allocate a completely infrastructure for SIP communication with several SIP proxies that play different roles among them. In this project MAX is able to place calls to the SBC. Currently, MAX calls to ”sip:caller1@64.131.109.38” from the user ”1001”. If an endpoint-user wants to answer a SIP call from MAX he/she has to register a SIP client to the ”64.131.109.38” domain with the name equals to ”caller1”.

2.6. Development steps: MAX Versions

- **MAX version 1**. At this point, MAX incorporated very basic functionality. It was able to recognize the user’s name among 10 different possibilities as well as the user’s city among 10 different values too. MAX was not able to place any call neither to store values in the database.

- **MAX version 2**. At this point, the VUI was changed to fix some errors of the previous version. MAX was fed with a dataset of thousands of person-names values. It also incorporated the algorithm to get the user’s address from the Amazon account. MAX was now capable to store values in the database so it now was able to remember users.

- **MAX version 3**. At this last point, MAX VUI’s was changed in order to be able to determine if the user is known. The capability to place standard calls, SIP calls and send SMS was added. At this point was ready to be able to save your life when an emergency was taking place.

3. FUTURE WORK AND CONCLUSSIONS

3.1. Future Work

There are some changes/additions that need to be done to get MAX’s full potential. These are sorted in order of relevance in the list showed below:

1. To change the VUI in order to be able to select what kind of communication the user wants to have (standard call, SMS, SIP call, one endpoint-user, several endpoint-users/ 911) by voice commands.

2. To change the VUI in order to be able to select what phone number/s the user wants to call/send an SMS by voice commands.

3. To change the VUI in order to make MAX able to store more user’s relevant information such as blood type or allergies.

4. To add an on-line platform in order that when a user registers it automatically rents a number from Twilio and it links to the user’s personal MAX.

5. To add analytics functions to manage users, behaviors and reported errors.

6. To add a complete guide of documentation to make users understand how MAX works and how to configure it.

7. Test and debugging to ensure no errors for the final user.

8. To do legally and business research to lunch the application safely and legally.

9. To upload the final and stable version to the Amazon skill market.

3.2. Conclusions

With this project, it has been achieved a necessary emergency assistant using modern technology resources such as cloud computing and Artificial Intelligence (AI). With very few funding it is possible to have a personal emergency assistant that can save user’s. When MAX is configured, the user only needs to have the Amazon Echo Dot plugged and connected to a WiFi signal in order to make it being permanently listening. Users do not need to move or even to have a phone available to ask MAX to help them. It is very useful for people with limited mobility, aged people, blind people or children among others.

4. ACKNOWLEDGMENTS

I would like to thank to the RTC lab for let me use its facilities and equipment, to the IIT for making this project possible, to the ClueCon 2018 developers where MAX was first presented and where I got useful feedback. A very special thanks to Carol for making a very needed mentorization role and for asking for funding from the RTC lab to buy the equipment.

5. REFERENCES

Nena. The 911 Asociation. Statistics.
Alexa Skill Kit documentation. Understand Custom Skills.
Alexa Skill Kit documentation. Create Intents, Utterances, and Slots.
Alexa Skill Kit documentation. Standard Built-in Intents.
AWS. Lambda. Developer Guide. What Is AWS Lambda?
AWS. AWS Identity and Access Management (IAM)
AWS. What is Amazon CloudWatch Logs?
AWS. Best Practices for Working with AWS Lambda Functions.
AWS. Amazon Resource Names (ARNs) and AWS Service Namespaces.
Apex (http://apex.run/)
TechTarget. AWS Tools. AWS. Alexa Skill Kit.
Wirecutter. What Is Alexa? What Is the Amazon Echo, and Should You Get One?
Wikipedia. Amazon DynamoDB.
w3schools. JSON Introduction.
Twilio. What is Twilio and How Does the Twilio API Work?
Twilio. TWIML^{TM} FOR PROGRAMMABLE VOICE.
Twilio. TWILIO’S REST APIS.
Twilio. Help Center. Sending and Receiving Limitations on Calls and SMS Messages.
Twilio. SMS pricing.
Node.js organization. Node.js v10.7.0 Documentation.