DOUGLAS MITSUO YAMADA YOSHIDA

EVERALDO APARECIDO GALIANO JUNIOR

VINICIUS GARCIA DA SILVA

SMART HOME PARA IDOSOS

São Paulo

2019

DOUGLAS MITSUO YAMADA YOSHIDA

EVERALDO APARECIDO GALIANO JUNIOR

VINICIUS GARCIA DA SILVA

SMART HOME PARA IDOSOS

Trabalho de conclusão de curso apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do título de Engenheiro de Computação.

Área de concentração:
Engenharia de Computação

Orientador:
Prof. Dr. Reginaldo Arakaki
Co-orientador:
Engenheiro Victor Hayashi

São Paulo
2019

Nome: YOSHIDA, Douglas Mitsuo Yamada. JUNIOR, Everaldo Aparecido Galiano. SILVA, Vinicius Garcia.

Título: Smart Home para Idosos.

Monografia (Trabalho de Conclusão de Curso) apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Bacharel em Engenharia na área de Engenharia de Computação.

Trabalho entregue para o departamento em:

Responsáveis

_____

Prof. Dr. Reginaldo Arakaki

(Orientador)

_____

Engenheiro Victor Hayashi

(Coorientador)

_____

Douglas Mitsuo Yamada Yoshida

(Orientado)

_____

Everaldo Galiano Aparecido Junior

(Orientado)

_____

Vinicius Garcia da Silva

(Orientado)

AGRADECIMENTOS

Ao professor Reginaldo Arakaki e ao Engenheiro Victor Hayashi, pela orientação e apoio durante todo o trabalho.

Aos familiares e a todos que colaboraram direta ou indiretamente na execução deste trabalho.

RESUMO

Este documento descreve o desenvolvimento de um sistema de casa inteligente voltado à manutenção da autonomia e independência do idoso. Com uma população mundial idosa cada vez maior, este trabalho preocupa-se em utilizar a tecnologia para prover saúde e bem-estar destas pessoas, reduzindo a necessidade de auxílio de terceiros. Para isso, foram cobertas duas jornadas: uma voltada à TV, representando entretenimento; e uma voltada a remédios, representando saúde. Para interface intuitiva com o usuário, foram criados um aplicativo e um *smart speaker.* São apresentados neste documento os conceitos básicos utilizados na base teórica que sustenta o projeto, como *IoT* e *smart homes*; as tecnologias envolvidas no sistema; as metodologias empregadas no desenvolvimento; os requisitos funcionais e não-funcionais que descrevem o escopo do projeto; a implementação; testes; resultados; e as considerações finais.

Palavras-chave: idoso; *smart home*, *smart speaker,* TV, caixa de remédios, *IoT*.

ABSTRACT

This document describes the initial development of a smart home system with a focus on maintaining the elderly autonomy and independence. With a growing elderly population, this project means to use technology to provide health and well-being to these people, diminishing the need for help from third-parties. To do so, two user journeys were covered: one aimed at the TV, supporting entertainment; and another aimed at medicines, supporting health. For a intuitive user interface, it was created a mobile app and a *smart speaker*. This document presents the basic theoretical concepts that ground the project, like IoT and smart homes; the technologies used in the system; the utilized development methodologies; the functional and nonfunctional requirements that describe the initial project scope; the implementation; tests; results; and final remarks.

Key-words: elderly; *smart home*, *smart speaker,* TV, *medicine box*, *IoT*.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

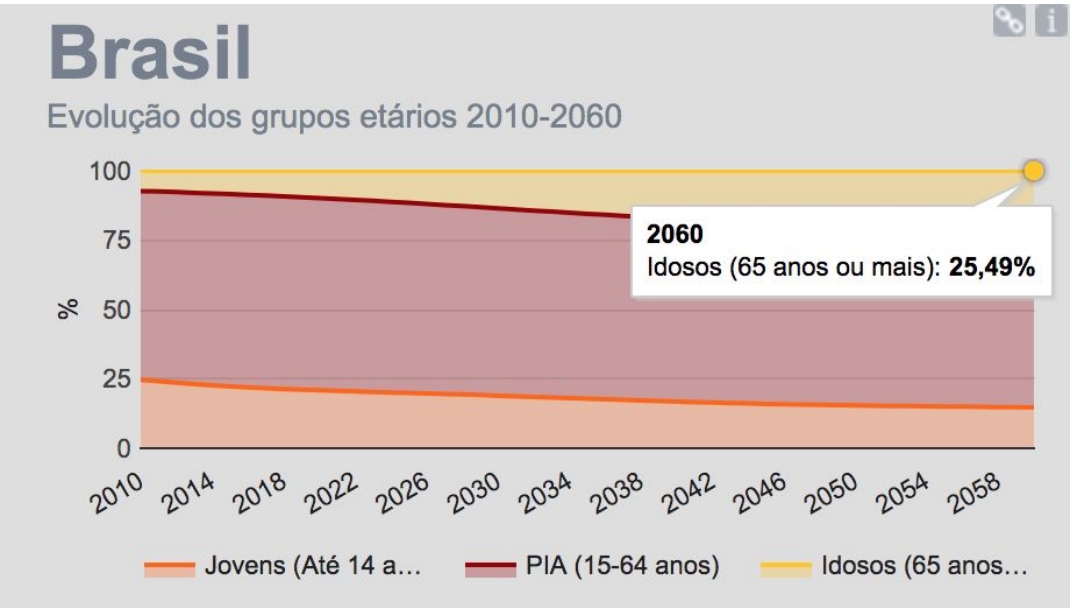
1 INTRODUÇÃO

Neste capítulo introdutório, são apresentados a motivação do projeto, sua justificativa, seu objetivo e a organização do trabalho.

1.1 MOTIVAÇÃO

O IBGE, Instituto Brasileiro de Geografia e Estatística, estima que até 2060 a população idosa brasileira, que hoje representa apenas 9.52% dos brasileiros, passará a ser 25.49% da população nacional[1], como mostra a projeção na figura 1.

Figura 1 - Evolução dos grupos etários brasileiros de 2010 a 2060



Fonte: (IBGE, 2019)[1]

Esta é uma tendência mundial e não são raros os casos de países que passam por transformações demográficas que alongam o topo da pirâmide etária. O governo japonês, por exemplo, anunciou em setembro de 2018[2], que 1 a cada 5 japoneses já possui 70 anos ou mais.

Essa crescente população de idosos vem exigindo que o Brasil e os demais países passem por adequações sociais, políticas e econômicas para atender os desejos e necessidades dessa parcela da população. Com a área tecnológica isto

não poderia ser diferente e novos desafios e soluções tecnológicas surgem para acompanhar essa transformação populacional da humanidade.

É uma preocupação constante desenvolver e aprimorar infraestruturas que mantenham a saúde e qualidade de vida das pessoas. Há, além disso, uma preocupação em manter a independência desta parcela populacional, provendo ferramentas que auxiliem nas tarefas domiciliares prejudicadas pelas dificuldades advindas da terceira idade.

Assim, essa demanda e os problemas associados de usabilidade e implementação são grandes motivadores deste trabalho.

1.2 JUSTIFICATIVA

Visando atender à vontade da população alvo de se manter independente, o que pode gerar conflitos familiares, o projeto se contrapõe a soluções invasivas, como o uso de câmeras e de visitas, deslocando preocupações rotineiras para um sistema automatizado e intuitivo que forneça maior autonomia. É proposta uma arquitetura para um sistema como tal, com possibilidade de expansão de funcionalidades.

Além disso, o projeto busca propor uma alternativa *open-source* de *smart speaker,* contornando limitações encontradas em produtos comerciais e fornecendo maior liberdade de implementação para melhor atingir os objetivos levantados.

1.3 OBJETIVO

Este trabalho descreve o desenvolvimento de um projeto voltado à população idosa que busca prover uma infraestrutura inteligente para auxiliar as rotinas básicas caseiras dessa parcela da sociedade, prezando particularmente pela independência das pessoas, sua saúde e qualidade de vida. Assim, foram priorizadas e implementadas duas jornadas, sendo elas uma caixa de remédios com lembretes, e uma TV automatizada. Além disso, para melhor usabilidade, foi desenvolvido um *smart speaker open-source*, permitindo o uso de linguagem natural

para interação com o sistema e um aplicativo de celular para fornecer uma segunda possibilidade de interface com características distintas da primeira.

1.4 ORGANIZAÇÃO DO TRABALHO

Este documento divide-se em oito seções. Primeiro, são apresentados o objetivo, a organização e a justificativa do projeto. Em seguida, são expostos aspectos conceituais que embasam a teoria utilizada durante a execução deste trabalho. A terceira seção apresenta as tecnologias empregadas na construção da aplicação final. Já na quarta seção, é exposta a metodologia de projeto utilizada. A quinta expõe a especificação básica de requisitos do sistema. A sexta seção apresenta o projeto e implementação do sistema proposto. Em sequência, a sétima seção discute os resultados obtidos. Por fim, a oitava e última seção apresenta as considerações finais.

2 ASPECTOS CONCEITUAIS

Nesta seção, são apresentados os aspectos conceituais que permeiam este trabalho. Primeiro, é apresentado o projeto Hedwig, base arquitetural deste projeto; em seguida, o conceito de *smart homes;* depois, um trabalho analisado a respeito da aceitação de idosos à *smart homes*; e, por fim, o conceito de *smart speaker open source*, um tema de estudo deste trabalho.

2.1 HEDWIG

Em 2017, o Engenheiro Victor Hayashi e alunos da engenharia elétrica da Escola Politécnica da USP desenvolveram uma solução de *smart home* como trabalho de conclusão de curso, o projeto Hedwig[3].

Figura 2 - Logo do projeto Hedwig

Fonte: (YASSUDA; MELO; POSSANI; HAYASHI, 2017)[3]

Para o presente trabalho de conclusão de curso o projeto Hedwig foi usado como base arquitetural para sua elaboração.

Na implementação da solução os conceitos e tecnologias chave foram IoT para integrar o sensoriamento e atuação na casa, inteligência artificial para aprendizado das rotinas do morador e arquitetura de micro-serviços na nuvem para garantir baixo acoplamento e alta escalabilidade dos serviços fornecidos. Esses temas serão abordados no tópico 3.

2.2 SMART HOMES

O artigo *A Knowledge-Driven Approach to Activity Recognition in Smart Homes* reforça a ideia de que, com o envelhecimento da população, smart homes vêm ganhando espaço ao permitirem maior independência[4].

Uma Smart Home é um ambiente residencial acrescido de sensores, atuadores e outros dispositivos conectados à internet, que coletam dados e auxiliam atividades do dia-a-dia.

Atualmente, a tendência é deixar de somente fornecer alertas e lembretes, para serem capazes de reconhecer atividades diárias. Porém, o artigo aponta diversos desafios, como a heterogeneidade das informações, e a falta de uma ordem definida para a realização de certas atividades, e define um modelo para superar essas dificuldades.

Este projeto combina aspectos das duas abordagens, fornecendo lembretes de programas de TV e de medicamentos de forma adaptável ao comportamento do usuário.

2.3 TRABALHO DE ACEITAÇÃO

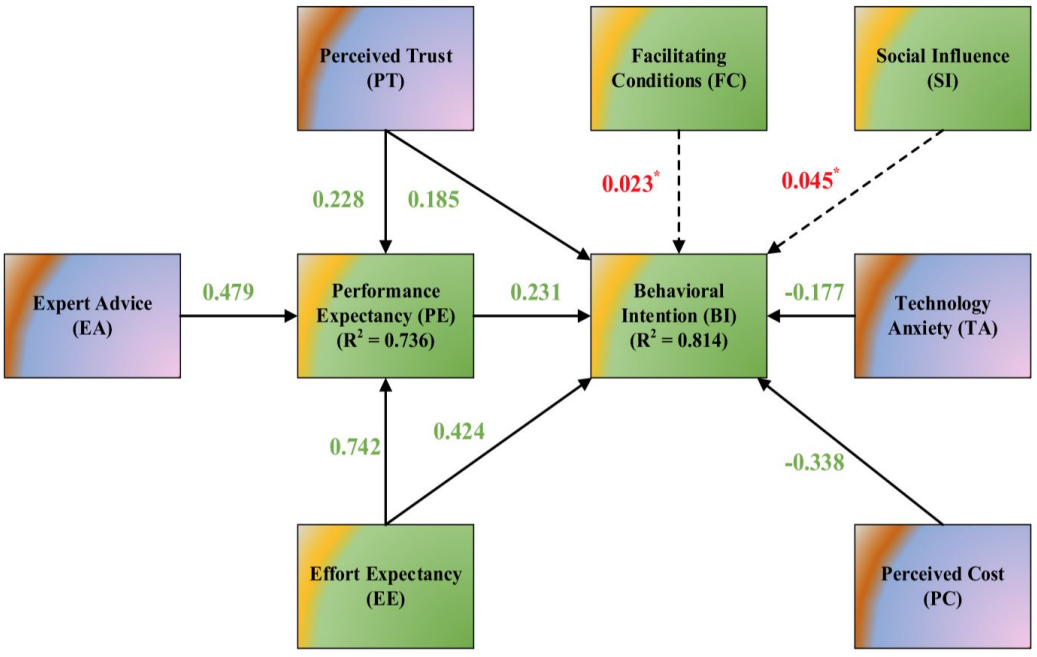
No artigo *Internet-of-Things and Smart Homes for Elderly Healthcare: An End User Perspective*[5], é feito um estudo para montar um modelo matemático das características que levam à aceitação do uso de Smart Homes.

De acordo com pesquisas anteriores os autores levantaram os fatores que mais influenciam a adoção da tecnologia (no artigo referido como Behavioral Intention). Eles são:

- Performance Expectancy (PE): Os benefícios que o usuário espera ter no uso da tecnologia.
- Effort Expectancy (EE): A facilidade de se aprender a utilizar a tecnologia.
- Perceived Cost (PC): O custo associado à adoção da tecnologia.
- Technology Anxiety (TA): O medo e apreensão ao se utilizar a tecnologia.
- Perceived Trust (PT): A confiança do usuário em utilizar a tecnologia, mesmo que seus dados estejam sendo coletados.
- Expert Advice (EA): A influência da opinião de profissionais de confiança do usuário.
- Social Influence (SI): A influência da opinião de amigos e familiares do usuário.
- Facilitating Conditions (FC): A crença do usuário de que haverá uma infraestrutura de suporte na adoção da tecnologia.

Os pesquisadores, então, entrevistaram potenciais usuários utilizando um roteiro de perguntas para medir a importância de cada uma das características formuladas. Com isso os autores montaram o modelo da aceitação com pesos, mostrado na figura 3.

Figura 3 - Modelo de aceitação



Fonte: (CHEN; NUGENT; WANG, 2012)[5]

O modelo demonstra que os aspectos mais importantes foram a Expectativa de Desempenho (PE), Expectativa de Esforço (EE).

2.2.3 SMART SPEAKER OPEN SOURCE

Para melhor fornecer autonomia ao público alvo, buscou-se melhorar a usabilidade permitindo interações em linguagem natural. Para esse fim, há os chamados *smart speakers* ou alto-falantes inteligentes, aparelhos conectados à internet capazes de receber comandos de voz e responder ao usuário, fornecendo funcionalidades diversas, como consultas a previsão do tempo, streaming de música, e até controle de iluminação. Como exemplos, há o Google Home e o Amazon Alexa. No entanto, os *smart speakers* comerciais são limitados aos serviços dos respectivos fabricantes e são restritos a ações reativas com o uso *wake words*, como o *'Ok Google'.* Dessa forma, visando ter maior liberdade no desenvolvimento das jornadas, com acionamento proativo e a possibilidade de escolha dos serviços básicos de *text to speech*, *NLU (natural language understanding)* e *speech to text* que melhor se adequarem, foi decidido desenvolver um *smart speaker open source*.

Diversas pesquisas e trabalhos já foram publicados voltados ao assunto de *smart speaker open source* e a segurança dos mesmos. O que se pode notar é que os trabalhos que buscaram formas alternativas de criar uma interface de voz apresentaram limitações como: não serem uma solução completa contendo os 3 serviços básicos de um *smart speaker*, suportar poucos idiomas e possuir hardwares adicionais a serem integrados aos dispositivos principais do mercado.

A biblioteca SpeechRecognition[6] traz uma forma de utilizar diversos serviços de reconhecimento de fala de uma forma simples. É possível escolher, via linha de comando, qual serviço se deseja utilizar para a transcrição de um áudio. A iniciativa, porém, limita-se somente ao primeiro serviço básico de *smart speaker*, sem incluir um módulo para a compreensão da fala nem um módulo para a transformação de uma resposta em fala.

Já o projeto *SnowBoy*[7] apresenta uma forma alternativa e com código aberto de criar uma *wake word*. Neste projeto, é possível treinar o *software* para reagir à uma certa palavra e ativar algum sistema subsequente. Nota-se aqui a preocupação em personalizar uma *wake word*, opção não disponível nos principais produtos do mercado. Assim como no caso supracitado, o projeto não contempla todas as etapas de um *smart speaker*.

O projeto *An Open Voice Command Interface Kit*[8] buscou criar uma solução alternativa para interface de voz de forma *open-source*, utilizando um *hardware* acessível com baixo custo e o *PocketSphinx* para a parte de *software* responsável pela transcrição da fala. Um dos empecilhos apresentados por este trabalho reside no fato de não suportar alguns idiomas, tal como o português, fato proveniente da limitação do *software* escolhido.

Do ponto de vista de segurança, o projeto *Detecting Surrounding Users by Reverberation Analysis with a Smart Speaker and Microphone Array*[9] indica uma preocupação com a segurança dos *smart speakers* do mercado, especialmente com um ataque chamado *DolphinAttack*. Este ataque consiste em usar frequências inaudíveis para enviar comandos para o dispositivo sem o consentimento do usuário. Devido à limitação dos aparelhos mais populares do mercado, a proteção contra este ataque fica dependente das ações dos fabricantes, uma vez que a arquitetura fechada desses *smart speakers* inibe grandes alterações por parte de desenvolvedores terceiros ou do usuário.

Em questão de privacidade, uma forma de resolver possíveis desconfianças em relação às principais empresas que desenvolvem *smart speakers* é o *Alias*[10], um dispositivo desenvolvido para ser usado nos *speakers* da *Amazon* e *Google*. Consiste em um *hardware* que bloqueia o que é ouvido pelo aparelho utilizando um ruído branco, com estabelecimento de uma nova *wake word*. Novamente, a arquitetura fechada dos produtos destas empresas impede que uma alteração visando privacidade seja feita, obrigando o usuário a recorrer às soluções como esta que utiliza um novo *hardware* cobrindo o *smart speaker*.

Assim, a criação de um *smart speaker open source* no contexto desse projeto busca criar uma interface baseada em linguagem natural intuitiva para o usuário e, ainda, resolver a problemática quanto à falta de flexibilidade dos dispositivos do

mercado por meio de uma solução de arquitetura aberta que permita adequar o dispositivo às condições que mais satisfazem o usuário, seja personalizando o aparelho, otimizando seu processo, suportando o idioma desejado ou buscando privacidade e segurança.

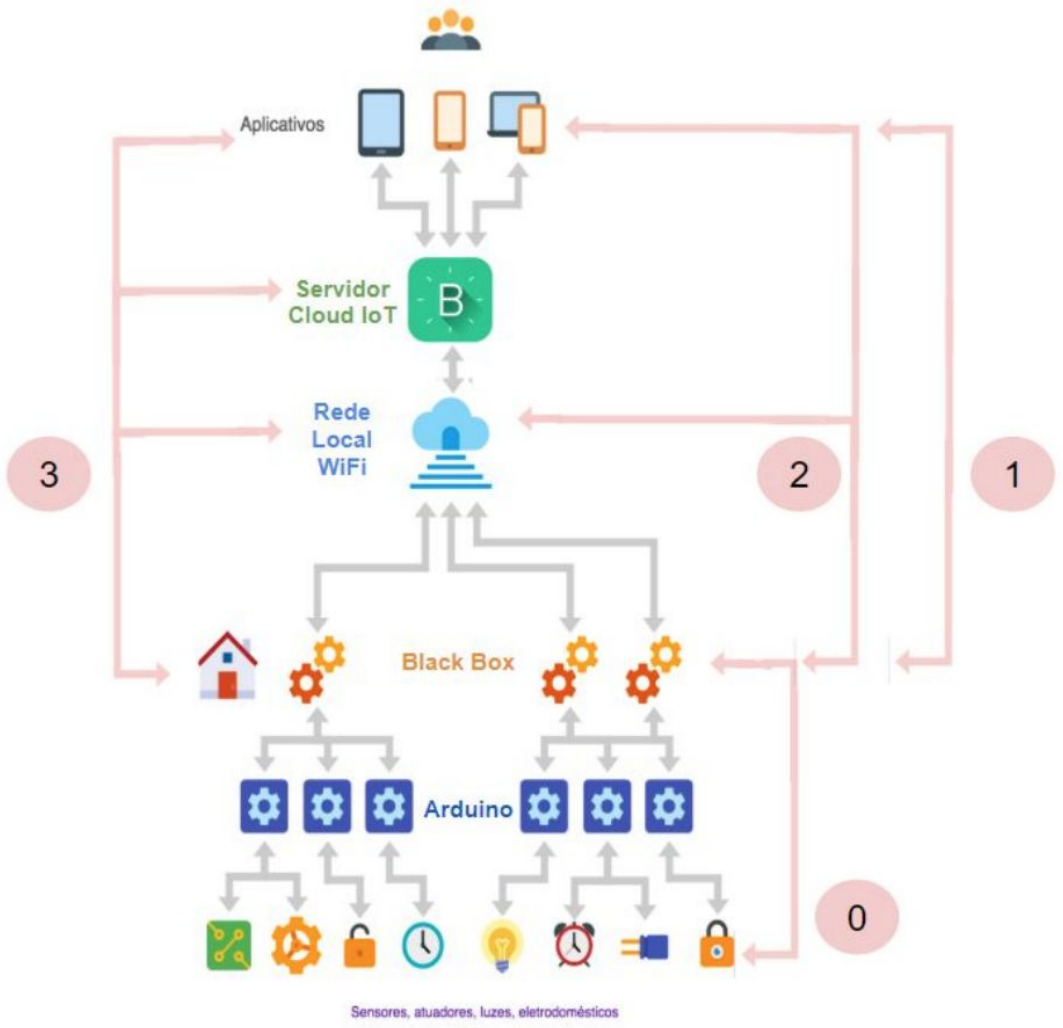
3 TECNOLOGIAS UTILIZADAS

Esta seção apresenta as principais tecnologias usadas na composição do projeto desenvolvido. São apresentados nesta seção o *Hedwig*, trabalho utilizado como base arquitetural deste projeto, os conceitos tecnológicos de *IoT*, as linguagens de programação utilizadas na etapa de desenvolvimento, os *hardwares* utilizados na implementação, os serviços utilizados no *smart speaker*, os serviços de nuvem empregados e o banco de dados escolhido para o projeto.

3.1 HEDWIG

O projeto *Hedwig*[3] descrito no tópico 2.1 foi orientado pelo Professor Dr. Reginaldo Arakaki, o orientador do presente projeto, e foi o projeto de formatura do atual co-orientador, Victor Hayashi. De tal forma a estrutura de *IoT* e microsserviços foi disponibilizada como base para o desenvolvimento deste projeto, na forma de um kit. A arquitetura geral do kit acelerador IoT *Hedwig* é apresentado na figura 4 abaixo.

Figura 4 - Arquitetura do kit acelerador IoT Hedwig[11]



Fonte: (HAYASHI)[11]

3.2 IOT

Internet of Things, ou, Internet das Coisas, é um conceito crescente e cada vez mais presente na sociedade moderna. A premissa básica deste conceito é a de “ter sensores inteligentes colaborando diretamente sem o envolvimento humano para entregar uma nova classe de aplicações” [12].

Figura 5 - Visão geral do *IoT*, mercados e sua integração

Fonte: (AL-FUQAHA; GUIZANI; MOHAMMADI; ALEDHARI; AYYASH, 2015)[12]

Pode-se dividir o mercado de *IoT* em vertical e horizontal. O primeiro engloba todos os objetos inteligentes, enquanto o segundo corresponde aos serviços analíticos e à computação ubíqua, como mostrado na figura 5.

A aplicação de tecnologias IoT é altamente abrangente e ultrapassa o âmbito domiciliar. Agricultura, medicina, transporte, indústria, entre muitas outras áreas, podem ser beneficiadas pela internet das coisas, como mostra também a figura 5.

Do ponto de vista arquitetural, pode-se dividir uma aplicação *IoT* em quatro camadas: sensoriamento, rede, serviço, e interface[13].

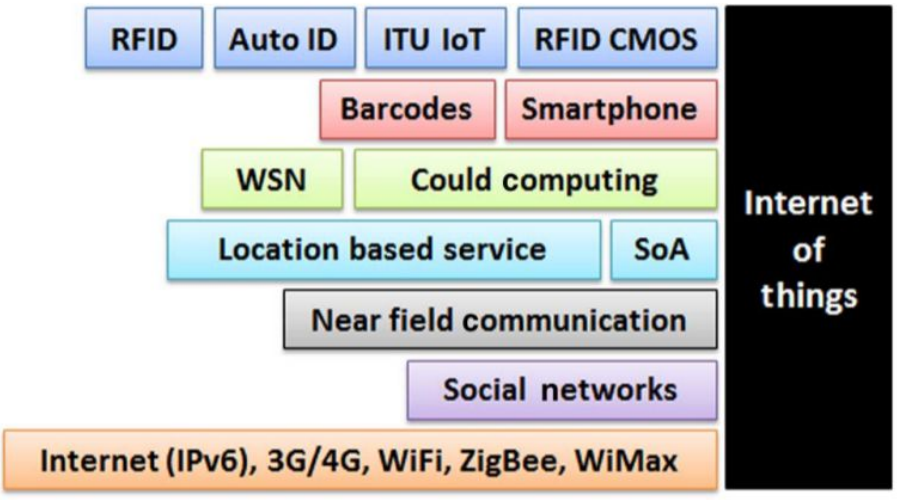
Tabela 1 - Camadas de uma aplicação IoT[13]

Camada	Descrição
Sensoriamento	Compreende os dispositivos de hardware para sensoriar e controlar o mundo externo e adquirir dados relevantes para aplicação.
Rede	Camada responsável por prover suporte na transmissão dos dados da aplicação via wireless ou cabeamento.
Serviço	Cria e gerencia os serviços da aplicação para satisfazer as necessidades do usuário.
Interface	Provê uma interface de interação entre aplicação e o usuário e/ou outras aplicações.

Fonte: Internet of Things in Industries: A Survey[13]

Para formar cada uma dessas camadas de uma aplicação *IoT*, diversas tecnologias podem ser usadas, como indicado na figura 6 abaixo em que são listadas algumas das tecnologias associadas a internet das coisas.

Figura 6 - Tecnologias envolvidas em IoT[13]



Fonte: (XU; HE; LI, 2014)[13]

3.3 LINGUAGENS DE PROGRAMAÇÃO

A seguir, são apresentadas as linguagens de programação empregadas no desenvolvimento do projeto. São elas: *JavaScript,* C e *Python.*

3.3.1 JavaScript

JavaScript é, primariamente, uma linguagem utilizada em navegadores web, mas com o interpretador *Node.JS* também é possível executar código no lado do servidor. Isso foi utilizado para criar os servidores das jornadas, devido à familiaridade do grupo com a linguagem e à sua agilidade.

Neste projeto, servidores *Node.JS* foram desenvolvidos para permitir armazenar os dados de uso da TV e consultar qual o canal provavelmente será assistido, através de requisições HTTPS a APIs implementadas com a *framework Express.JS*. Um outro servidor similar é usado para armazenar e consultar lembretes referentes ao uso de remédios do usuário. O hub (sistema central do projeto) também é um servidor que emprega estas tecnologias.

Um aplicativo para cadastro de lembretes foi desenvolvido utilizando *TypeScript*, que torna o código mais organizado ao oferecer tipagem sobre o *JavaScript* comum. Foi utilizada a biblioteca *React Native*, que gera elementos de interface nativos *iOS* e *Android*, com um código-fonte em comum que usa *React*, uma biblioteca que permite usar tags JSX para construção de componentes.

3.3.2 C

C permite com mais facilidade o uso de operações de baixo nível. Neste projeto, a linguagem foi utilizada para programação de placas através da IDE Energia, incluindo o mapeamento de sensores e leds da caixa de remédios. Novamente, houve escolha desta linguagem principalmente devido à familiarização tecnológica do grupo.

3.3.3 Python

Python possui o diferencial de possuir uma grande quantidade de bibliotecas e integração com diversos serviços *web*, o que levou à sua escolha como tecnologia utilizadas em alguns pontos do projeto, como os serviços de processamento de linguagem natural para o speaker, e no *Raspberry Pi*, onde foi usada uma biblioteca de processamento de som.

3.4 HARDWARES

Neste tópico, são apresentados os principais *hardwares* utilizados no projeto. São eles: *Raspberry Pi*, o módulo de comunicação do projeto *Hedwig,* utilizado como acelerador e a placa *tiva launchpad*.

3.4.1 Raspberry Pi

O *Raspberry Pi* é um computador contido em uma única placa muito utilizado pela comunidade *Maker* para implementar projetos embarcados. O seu SoC Broadcom BCM2837 inclui quatro núcleos de processamento ARM Cortex-A53 com clock 1,2 GHz[13]. Ele tem várias portas seriais e USB além de pinos de GPIO e comunicação bluetooth e wifi. Ele é frequentemente utilizado com uma distribuição Linux como OS.

Figura 7 - Raspberry Pi

Fonte: (RASPBERRY PI FOUNDATION)[14]

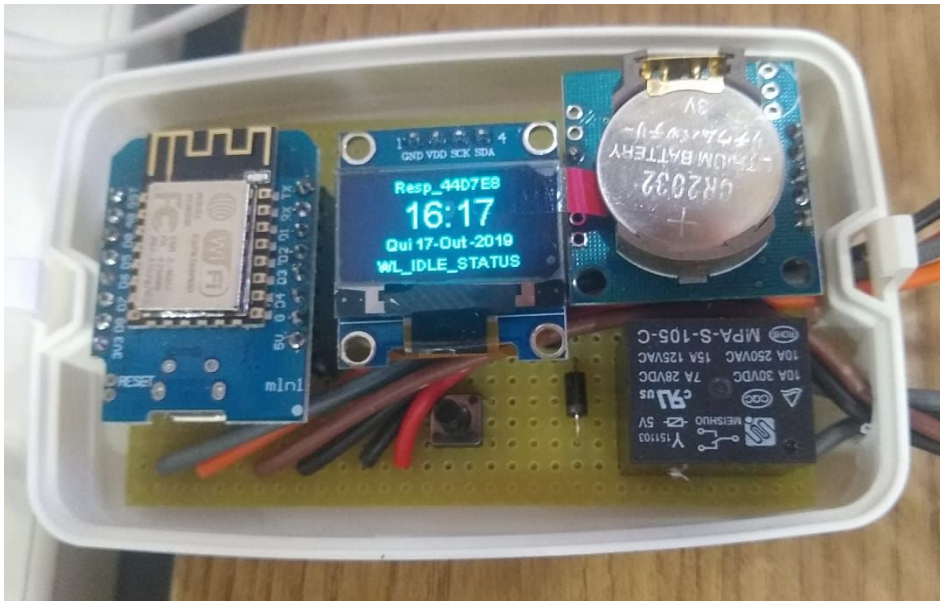
A escolha do raspberry pi como hardware do *smart speaker* e do *hub* se deu principalmente à familiaridade do grupo com ele, bem como ao fato de ser um hardware que possibilita a utilização do sistema *Linux* de maneira simples e pouco custosa.

3.4.2 Módulo de comunicação

O dispositivo caixa-preta do kit acelerador *IoT Hedwig* foi utilizado para realizar a comunicação de vários módulos de uso específico em um cômodo com a central da casa. Ele é feito baseado no microcontrolador ESP8266 para utilizar as funcionalidades como *access point Wifi*, mas também se comunica por meio de serial. Como parte do conjunto caixa-preta, tem-se um RTC (*Real Time Clock*) para sincronização de tempo offline, um display OLED que além de informações de data e hora permite consultar o IP do módulo, e um relé para controle de dispositivos de potência.

É possível controlá-lo pela rede local ou pela internet usando o serviço *Blynk*. O coorientador do projeto e coautor do *Hedwig* disponibilizou um manual com essas e outras possibilidades, incluindo integração com Arduino e Google Assistant.[11]

Figura 8 - Módulo de comunicação

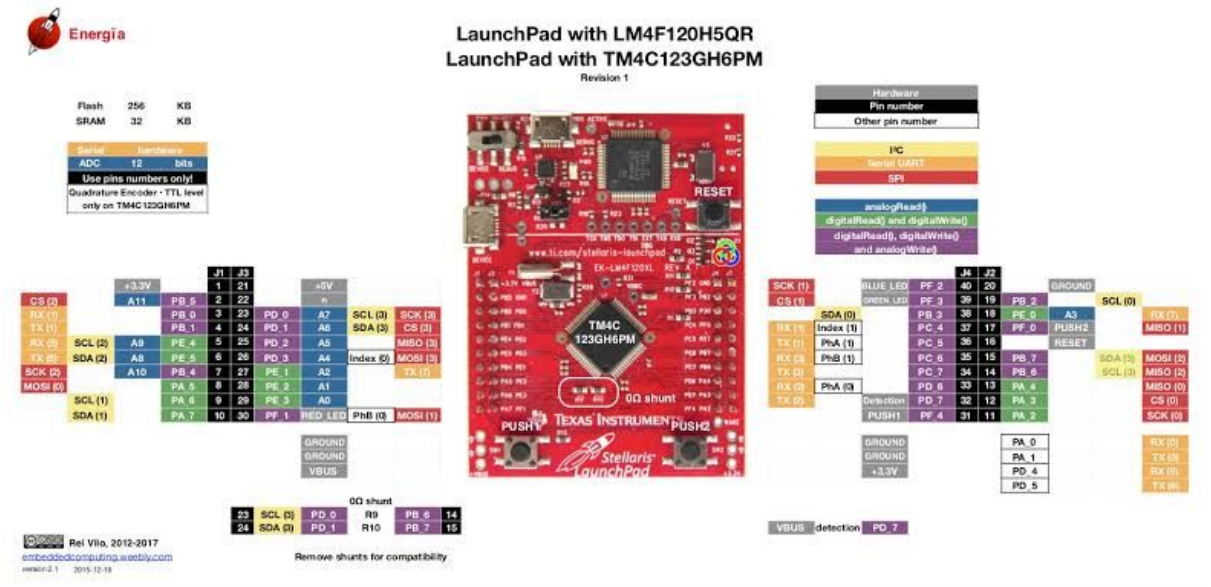


Fonte: (Própria, 2019)

3.4.3 Tiva launchpad

O Tiva launchpad é uma placa desenvolvimento baseado em um microcontrolador ARM Cortex-M4[14]. Por ser compatível com o *Energia*, uma *IDE* baseada no *Arduino*, é fácil e rápido prototipar soluções embarcadas. Por ter mais de 30 pinos I/O com capacidade de interrupção e múltiplos canais seriais, ele é ideal para controlar múltiplos atuadores e ler muitos sensores.

Figura 9 - Tiva launchpad



Fonte: (TIVA)[15]

3.4.4 Módulo IR

É similar ao módulo de comunicação descrito em 3.4.2, com o mesmo microcontrolador e também com as mesmas possibilidades para ser controlado remotamente. Porém, possui um emissor e um receptor IR, usado na jornada da TV.

3.5 SMART SPEAKER

As tecnologias analisadas e utilizadas no *smart speaker open source* desenvolvido podem ser divididas em 3 categorias: *speech to text* (3.5.1), *natural language understanding* (3.5.2), e *text to speech* (3.5.3).

3.5.1 Speech To Text

Serviços de *speech to text* são serviços cujo objetivo é transcrever um arquivo de áudio em texto. Neste trabalho, foram analisadas e utilizadas as ferramentas *Google Speech to Text* [16] e *Watson Speech to Text* [17].

3.5.2 NLU

Serviços de *natural language understanding* possuem inteligência suficiente para analisar uma frase escrita, interpretá-la e determinar possíveis intenções do usuário a partir desta frase, além de fornecer uma resposta à sentença analisada. Assim, foram analisados e utilizados o *Google Dialog Flow* [18] e o projeto *open source Rasa* [19].

3.5.3 Text-To-Speech

Serviços de text to speech são serviços cujo foco é transformar um texto em um arquivo de áudio. Neste trabalho, foram analisados e utilizados o *Google Text to Speech* [20] e o *Amazon Polly* [21].

3.6 CLOUD

O processamento de hábitos e lembretes é feito remotamente, retirando a necessidade de maior desempenho nos aparelhos locais e permitindo *backup* dos dados. Nota-se, no entanto, que localmente uma parte dos dados é mantida para permitir funcionamento offline.

3.6.1 Heroku

Heroku é uma plataforma cloud, ou seja, que fornece recursos computacionais remotos. Esta plataforma permite a execução de servidores desenvolvidos em diversas linguagens e possui um plano gratuito que atende aos propósitos de prova de conceito do projeto. Entretanto, este plano é limitado em termos de desempenho, já que desabilita a máquina remota enquanto não estiver em uso, causando um atraso inicial maior, uma vez que é necessária a reinicialização do serviço após certo tempo de inatividade.

3.7 BANCO DE DADOS

O PostgresSQL foi utilizado como o sistema Gerenciador de Banco de Dados Relacional do projeto. A escolha deste serviço foi devido à sua disponibilidade no *Heroku*, serviço de hospedagem escolhido para este projeto. O Postgres permite simples integração com o projeto através da ferramenta *heroku-cli* e através de bibliotecas para *Node.JS*. É utilizado para armazenamento das rotinas de TV e também para os lembretes de medicamentos.

4 METODOLOGIA DO PROJETO

Esta seção descreve a metodologia de projeto empregado na elaboração deste trabalho. O capítulo divide-se em concepção, projeto, programa de residência de software do Bradesco, implementação e testes.

4.1 CONCEPÇÃO

O tema inicial foi decidido em reuniões entre os integrantes do grupo, onde foram levantados diversos problemas a serem solucionados por meio de tecnologias. Houve uma convergência de ideias no tópico saúde e bem estar de pessoas idosas, pois este foi um tema presente na vida de alguns dos membros que tiveram casos pessoais de familiares idosos com dificuldades em algumas rotinas diárias e acidentes domésticos que poderiam ter sido evitados. Houve, assim, a ideia de elaborar um sistema que facilite a vida desse público alvo.

Com o tema e orientadores decididos, foi feita uma reunião, elicitando as possíveis jornadas a serem atendidas pelo sistema. O projeto tomou forma de uma *smart home* para acessibilidade, com aprendizado de hábitos e lembretes.

4.2 PROJETO

O primeiro passo para o planejamento do projeto consistiu em estudar o estado da arte na área de *IoT*, *smart homes* e tecnologias voltadas para idosos. Viu-se que esta uma área cada vez mais explorada e a causa cada vez mais preocupante, visto o gradativo aumento da população idosa no Brasil e no mundo.

Em seguida, reuniões entre os membros do grupo, o orientador e coorientador levantaram algumas possíveis jornadas a serem abordadas como alvo do sistema. Após muita discussão, algumas ideias foram descartadas e outras lapidadas.

As jornadas selecionadas foram definidas a partir de uma rotina exemplo do seu desenrolar. Após isso, foram definidos os requisitos funcionais e não-funcionais de cada uma delas.

Com as jornadas e seus requisitos definidos, foram determinadas as atividades necessárias para a realização do projeto, o que inclui estudos de tecnologias, definição de arquiteturas, e implementação. As atividades foram distribuídas entre os integrantes e realizadas por método ágil iterativo, divididas em sprints com reuniões com os orientadores no fim de cada ciclo, havendo validação de atendimento de requisitos, com redirecionamentos ou descartes conforme o caso. Para cada *sprint*, foi feita uma divisão de tarefas entre os membros do grupo para haver paralelização de esforços de acordo com afinidade técnica, disponibilidade e interesse.

4.3 PROGRAMA DE RESIDÊNCIA DE SOFTWARE DO BRADESCO

O programa de residência de software do Bradesco Inovabra é uma modalidade onde o aluno recebe apoio financeiro, técnico e material para se dedicar ao projeto do TCC, com créditos válidos para estágio. Sempre respeitando as diretrizes acadêmicas que envolvem o projeto de conclusão de curso: ou seja, o departamento, o professor coordenador e professor orientador, a residência de software que traz apoio ao aluno em termos de domínio de problema e de aplicação, orientação técnica complementar com experiência dos especialistas da indústria, além da ajuda com a bolsa auxílio e aquisição de eventuais materiais para o projeto. Essa modalidade tem a virtude de apoiar o aluno na realização de um projeto de seus sonhos. Contribui para as metas da disciplina e para o currículo do departamento e também conhecer aspectos industriais que fundamentam o início de carreira do profissional, na fase final da sua graduação.

4.4 IMPLEMENTAÇÃO

A base da infraestrutura *IoT* do sistema tem como acelerador o projeto *Hedwig*, apresentado na seção 2.1 e descrito tecnicamente na seção 3.1. além de uma parceria com o Bradesco e sua área de tecnologia, a antiga Scopus.

A implementação foi feita paralelizando o desenvolvimento do *smart speaker* e das duas principais jornadas do projeto: a jornada da TV e a jornada do remédio.

Assim, em formatos de sprints, foi-se desenvolvendo *software* e *hardware*, para somente então fazer a integração, seguindo uma abordagem *bottom-up*.

Após o desenvolvimento das rotinas separadamente, foi desenvolvido o hub responsável por integrar as jornadas e centralizar a inteligência geral do projeto.

O aplicativo para cadastro de medicamentos e lembretes de TV foi implementado em paralelo ao hub, inicialmente usando endpoints para bancos temporários, para então ser integrado com o hub.

4.5 TESTES

Após o processo de implementação, foram realizados testes para confirmação do funcionamento correto do sistema e averiguação dos resultados, de acordo com os requisitos funcionais e não funcionais elaborados, descritos mais adiante na seção 5. Os testes foram elaborados de forma a cobrirem todos os fluxos planejados para o sistema e descritos posteriormente na seção 6.

5 ESPECIFICAÇÃO DE REQUISITOS DO SISTEMA

Dado a contextualização do problema apresentado, neste capítulo são desenvolvidas as especificações de requisitos do sistema, apresentando as especificações da jornada da TV e do remédio e, consequentemente, as demais aplicações do sistema que auxiliam nestas jornadas, como o aplicativo e o *smart speaker*.

5.1 JORNADA DA SMART TV

Esta jornada descreve o caso de rotinas envolvendo entretenimento. O usuário final é um usuário idoso sem grandes afinidades com Smart TVs e, por isso, a jornada trata da acessibilidade da tecnologia para este público.

5.1.1 Descrição

1) Gertrudes ganhou uma Smart TV;
2) Ela instala um módulo do sistema perto de sua nova TV e seu aparelho conversor de TV;
3) Por alguns dias, ela assiste sua novela às 21h no canal X;
4) A inteligência do sistema detecta o hábito;
5) Certo dia, Gertrudes vai à sala assistir à sua novela como de costume;
6) O sistema detecta sua presença na sala para cumprir seu hábito e pergunta à ela se deseja assistir à novela;
7) Gertrudes responde que sim, a TV é ligada automaticamente e trocada para o canal correto da novela;

5.1.2 Requisitos

A seguir, são enumerados os requisitos funcionais e não funcionais da jornada da TV.

5.1.2.1 Requisitos funcionais

Tabela 2 - Requisitos funcionais da jornada da TV

Requisito	Nome	Descrição
RF1	Comando por Voz ou App	Comando (trocar canal, ligar/desligar TV, mudança de volume) da TV por voz via *smart speaker* ou pelo aplicativo de celular.
RF2	Aprendizado de hábito	Comandos automatizados de TV baseado na rotina do usuário.

5.1.2.2 Requisitos não-funcionais

Tabela 3 - Requisitos não-funcionais da jornada da TV

Requisito	Nome	Descrição
RNF1	Acessibilidade	Usuário deve conseguir comandar a TV intuitivamente via aplicativo ou por linguagem natural de forma que não exija conhecimentos tecnológicos avançados do usuário.
RNF2	Usabilidade	Sistema deve ser capaz de automatizar a rotina do usuário a partir de sua instalação de forma independente da ação direta do usuário.

5.2 JORNADA DO REMÉDIO

Essa jornada descreve o caso geral de rotinas envolvendo remédios. O usuário final são idosos que necessitam tomar remédios esporadicamente ou regularmente. Por vezes, pode ser difícil lembrar-se de tomar remédios e mesmo organizar-se para tomá-los, não sendo raro ocorrer situações onde um remédio é tomado mais vezes do que o necessário em um dia, seja esquecido de ser tomado, ou mesmo tomar remédios por engano caso haja mais de um. Uma entrevista foi feita com possíveis usuários do sistema que comprovam esse problema e se encontra no apêndice deste documento. Por ser uma tarefa que pode ter um papel de extrema relevância na vida do usuário, é importante que isso seja feito com

cuidado e organização. Assim, a proposta desse projeto é a de criar uma caixa de remédios inteligente capaz de auxiliar o usuário nesta organização.

5.2.1 Descrição

1) Antônio possui diversos medicamentos que precisa tomar em diferentes horários do dia e diferentes dias da semana. Assim, Antônio tem o hábito de separar seus remédios por dia e horário.
2) Antônio separa seus remédios nos compartimentos da Caixa de Remédios Inteligente por dia e horário.
3) O sistema detecta o hábito de Antônio e registra os remédios que ele deve tomar e os devidos horários.
4) Todo dia, no horário específico de cada medicamento, Antônio vê o LED de interface da caixa de remédio que o lembra se ele já tomou ou não aquele medicamento.
5) Após tomar o remédio o sistema registra o evento e pergunta se ele adicionará uma nova dose.
6) Antônio responde que sim e registra novamente o remédio no sistema.
7) Certo dia, Antônio esquece de tomar seu medicamento e fica assistindo TV.
8) O sistema avisa Antônio que ele esqueceu de tomar seu remédio e assim ele o faz.

5.2.2 Requisitos

A seguir, são enumerados os requisitos funcionais e não funcionais da jornada do remédio. Para criar estes requisitos, utilizou-se como base teórica o trabalho de aceitação descrito na seção 2.3. Nele, é concluído que os fatores mais relevantes para a aceitação de um idoso à um sistema de *smart home* são a Expectativa de Desempenho e a Expectativa de Esforço. Assim, os requisitos funcionais buscam satisfazer a Expectativa de Desempenho, enquanto os requisitos não funcionais buscam satisfazer a Expectativa de Esforço.

5.1.2.1 Requisitos funcionais

Tabela 4 - Requisitos funcionais da jornada do remédio

Requisito	Nome	Descrição
RF3	Programação de remédio	Usuário deve ser capaz de programar no sistema um horário e dia específicos para um certo tipo de remédio incluído em um dos compartimentos da caixa de remédio. As informações devem poder ser passadas tanto via aplicativo de celular como por linguagem natural via *smart speaker*.
RF4	Lembrete de remédio	Sistema deve lembrar o usuário de tomar o remédio caso o horário já tenha sido atingido e o remédio não tenha sido tomado. O lembrete deve ser feito primeiro via led de sinalização da caixa de remédios e depois, caso seja necessário, via *buzzer* e *smart speaker*. Ao tomar o remédio, o sistema deve ser capaz de desativar todas as informações de alerta.

5.1.2.2 Requisitos não-funcionais

Tabela 5 - Requisitos não-funcionais da jornada do remédio

Requisito	Nome	Descrição
RNF3	Acessibilidade	Usuário deve conseguir programar um remédio na caixa de remédios de forma intuitiva, com poucos comandos e sem necessidade de conhecimentos tecnológicos avançados.
RNF4	Disponibilidade	Sistema deve ser capaz de lembrar o usuário de tomar seu remédio caso seja necessário independentemente de conexão com a internet.

6 PROJETO E IMPLEMENTAÇÃO

Este capítulo descreve o processo de projeto e implementação do sistema proposto. É especificado na seção 6.1 o *smart speaker* criado, a implementação da jornada da TV na seção 6.2 e a implementação da jornada do remédio na seção 6.3.

6.1 SMART SPEAKER

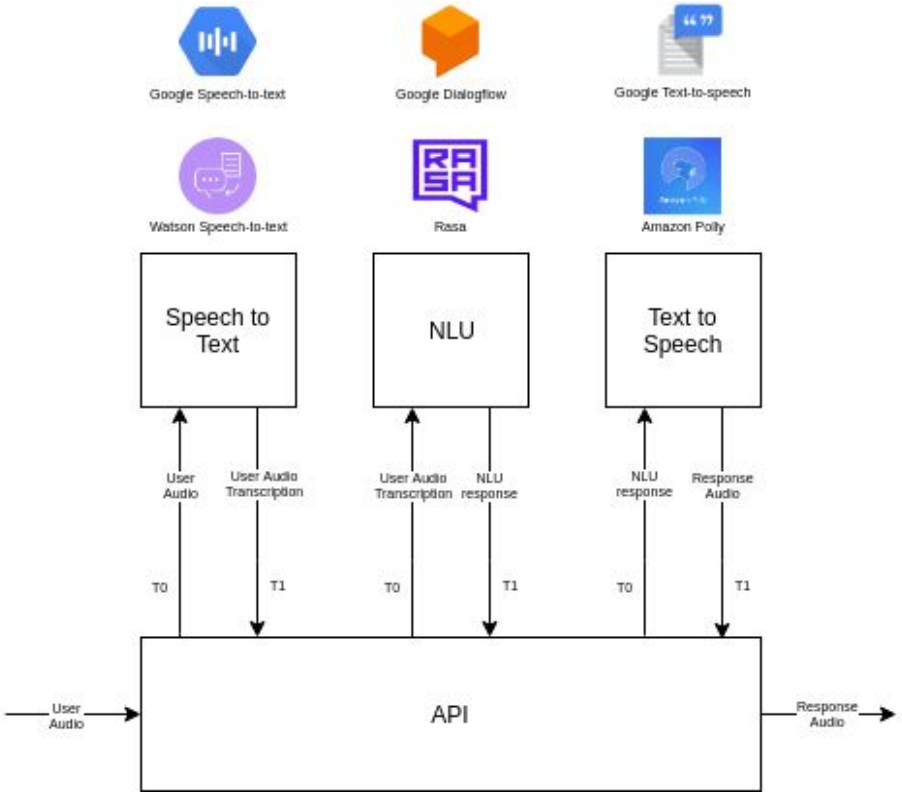
O *smart speaker* construído para esse projeto foi um *smart speaker open source.* Um estudo minucioso foi realizado e descrito melhor na seção 7.1.5 para definir a necessidade de o dispositivo ser *open source* e quais serviços ele utilizaria para seu funcionamento.

6.1.1 Software

O *smart speaker* disponibiliza uma API executando localmente a função de, a partir de uma chamada recebida, analisar um áudio de entrada, devolver as intenções do usuário e um arquivo de áudio com uma possível resposta.

É possível na chamada à API escolher o serviço que melhor se adequa às necessidades da situação. A figura 10 apresenta as alternativas disponíveis para as funcionalidades citadas do speaker. A melhor opção, levando em consideração aspectos quantitativos e qualitativos, segundo a seção 7.1.5, é a combinação *Google Speech-to-Text*, *Rasa* e *Polly*. Entretanto, devido à afinidade técnica, o grupo optou por utilizar o *DialogFlow* como serviço de NLU.

Figura 10 - API *smart speaker*



Fonte: (Própria, 2019)

Além disso, a API também é preparada para receber um texto, interpretá-lo via serviço de NLU e produzir uma resposta em áudio de voz.

6.1.2 Hardware

O hardware do *smart speaker* é feito utilizando um *Raspberry Pi* conectado a uma placa de som USB e um microfone comum. O *Raspberry* executa um programa feito em *python* para capturar um *stream* de áudio a partir do USB. Este áudio alimenta a API descrita na seção 6.1.1, onde ele será interpretado e uma resposta será produzida a partir dele.

Os dispositivos utilizados no *smart speaker* são encapsulados em uma carcaça produzida em uma impressora 3D, exibida na figura 11 abaixo e criada pelo designer Santiago Sulzbeck Villalobos.

Figura 11 - *Smart speaker*

Fonte: (Própria, 2019)

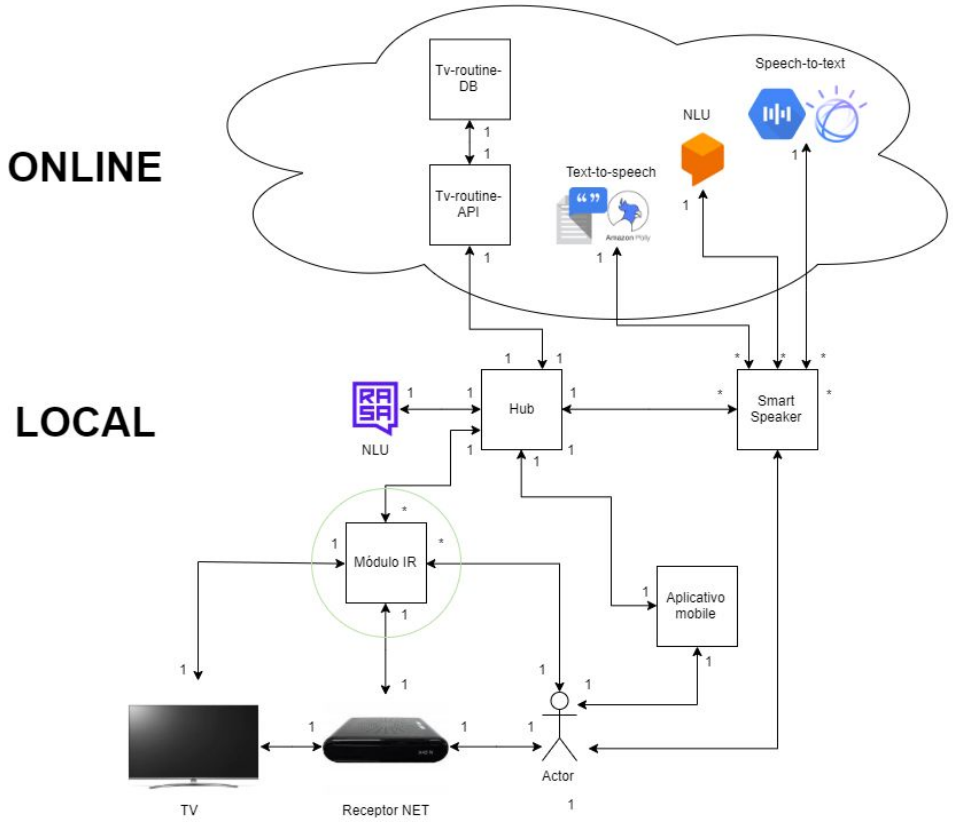
6.2 JORNADA DA TV

Esta seção descreve o projeto e implementação referente a jornada da TV definida no tópico 5.1. Primeiro, é descrita a arquitetura da jornada, seguida pela descrição dos módulos físicos utilizados nela, a API criada para manipulação dos dados e, por fim, a inteligência incluída no hub centralizador do sistema.

6.2.2 Arquitetura

A arquitetura da jornada da TV é ilustrada na figura 12 abaixo.

Figura 12 - Arquitetura da Jornada da TV



Fonte: (Própria, 2019)

Como se pode notar, a arquitetura distribui-se de forma local e online. Na parte local, há o *smart speaker*, o hub, o Módulo IR, a TV e o Receptor NET. Já na parte *online,* há a API, o banco de dados, o serviço de NLU, o serviço de text to speech e o serviço de *speech to text*.

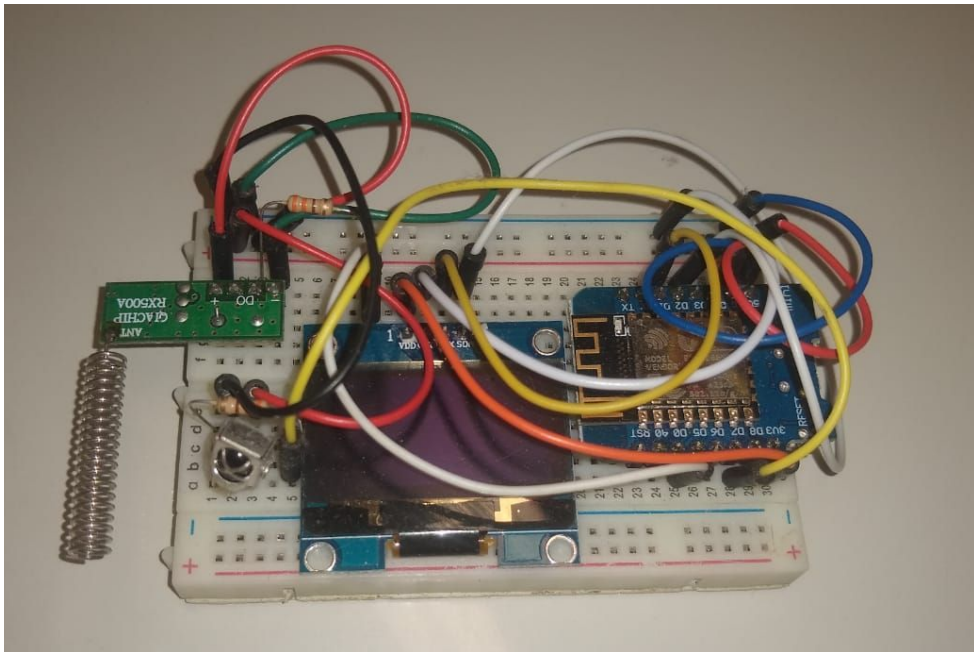
6.2.3 Módulos

Esta jornada conta com dois módulos: o módulo IR, responsável por captar e enviar sinais de infravermelho e o HUB, centralizador do sistema e gerenciador desta jornada.

6.2.3.1 Módulo IR

O módulo IR é responsável por captar os sinais emitidos pelo controle do aparelho de TV e enviar sinais para o receptor. Assim, cada comando enviado pelo usuário é registrado por esse módulo e devidamente repassado ao Hub. Por outro lado, este módulo pode, também, ser acionado para enviar um comando para o receptor NET, alterando o canal, volume, etc.

Figura 13 - Módulo receptor IR



Fonte: (Própria, 2019)

6.2.3.2 Hub

Nesta jornada, o módulo hub é responsável por servir como um intermediário entre o módulo IR e a API *cloud*. Além disso, ele contém a inteligência da jornada que determina o momento de avaliação da rotina, além do momento de armazenamento dos dados obtidos do usuário.

6.2.4 API

A API desta jornada tem duas funções primárias: gerenciar os dados do banco de dados e prover informações a respeito da rotina a partir dos dados do banco.

Na função de gerência, a API é programada para ser capaz de ler dados providos pelo HUB, captados pelo módulo IR, interpretá-los e transformá-los em dados próprios do sistema.

A tabela 6 abaixo ilustra um exemplo de como funciona o tratamento de dados da API.

Tabela 6 - Tratamento de dados da API da jornada da TV

1559517547, IRRX, 3772829743	2019-06-02T23:19:07.000Z, SAM_VOLUMEDOWN,
1559516462, IRRX, 3782893727	2019-06-02T23:01:02.000Z, NET_SIX

A API recebe o dado no formato “*time stamp, type, code”*, e o converte em uma data legível e na tradução do código IR obtido para o controle da TV. A partir disto, é possível interpretar os dados para compreender qual canal está sendo assistido, bem como saber se a TV foi ligada ou desligada.

6.2.5 FLUXOS

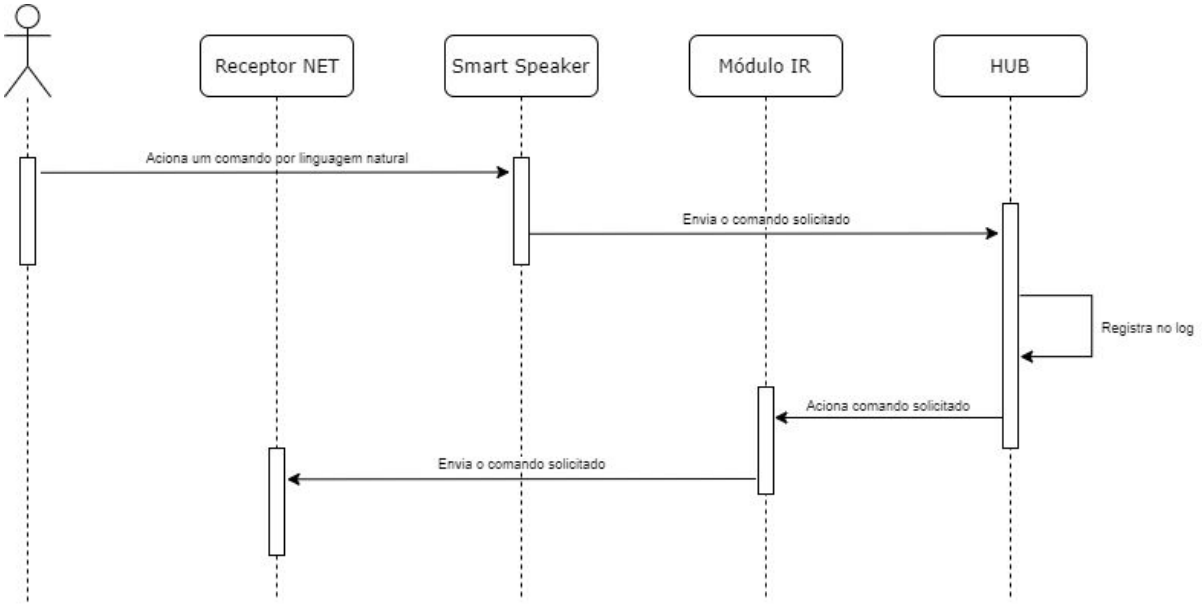
A seguir, são apresentados os fluxos principais correspondentes à jornada da TV, sendo eles o envio de comandos por linguagem natural e app, a captura de comandos, a interpretação de comandos diária e a análise de rotina diária.

6.2.5.1 Envio de comandos por linguagem natural e app

Este fluxo descreve o processo de envio de um comando para a televisão por meio do *smart speaker*, utilizando linguagem natural, e por meio do aplicativo de

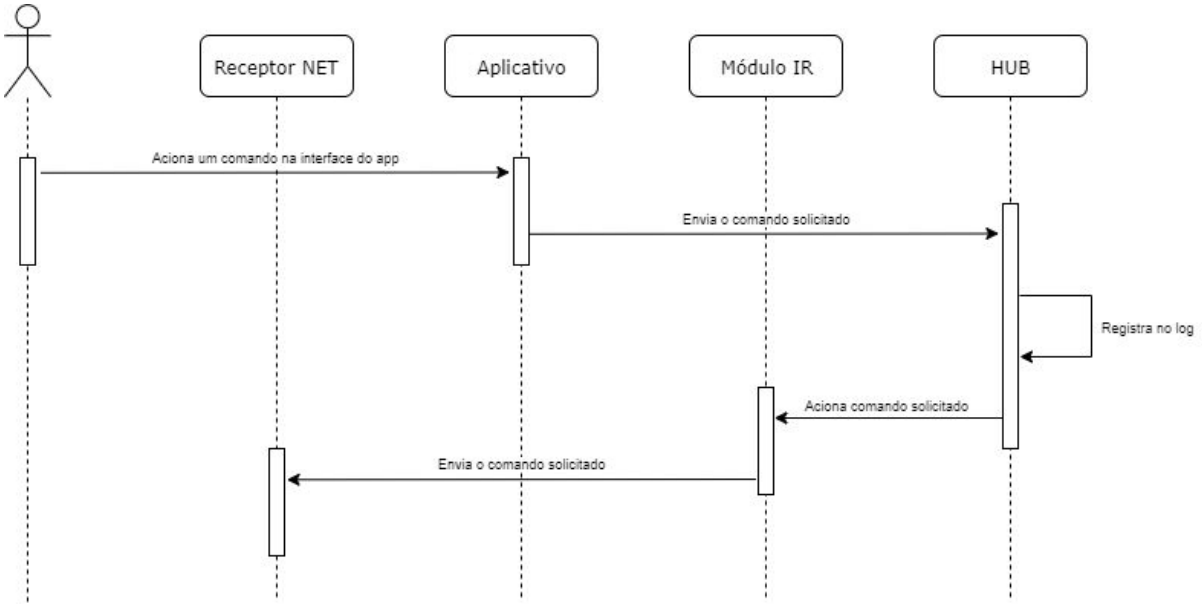
celular. Os diagramas de sequência que representam este fluxo se encontram nas figura 14 e 15 abaixo.

Figura 14 - Fluxo de comandos por linguagem natural



Fonte: (Própria, 2019)

Figura 15 - Fluxo de comandos por linguagem app



Fonte: (Própria, 2019)

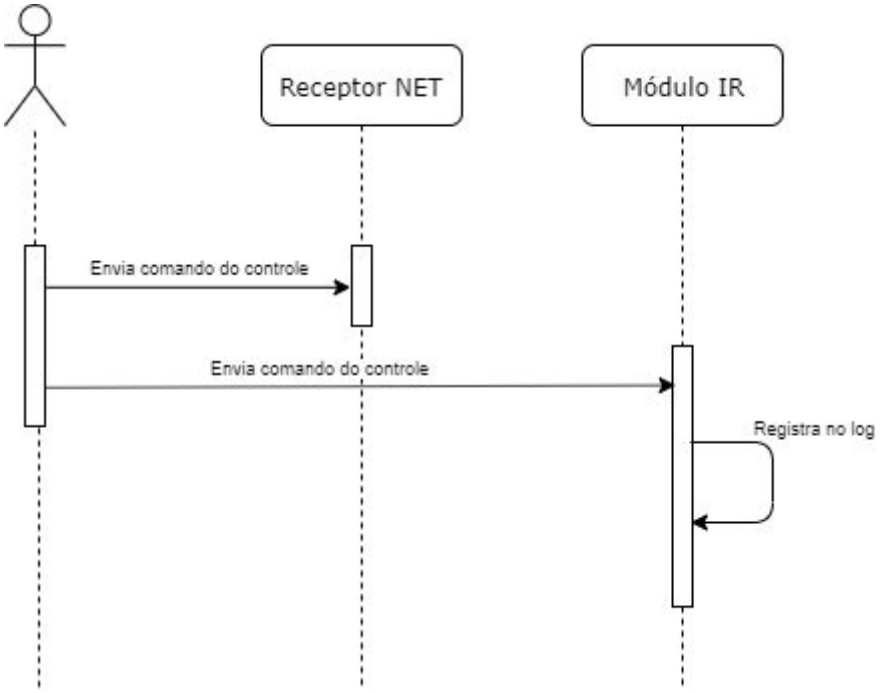
Para enviar um comando por linguagem natural, o usuário interage diretamente com o *smart speaker*, acionando-o e dizendo o comando desejado.

Após transcrever e interpretar o comando, o *smart speaker* envia o comando solicitado ao hub que, por sua vez, registra o comando e o aciona por meio do módulo IR, responsável por, finalmente, dar a ordem ao receptor NET do sistema. Analogamente, para enviar o comando via aplicativo, o usuário interage com o aplicativo *mobile*, que sucederá o fluxo assim como o *smart speaker*.

6.2.5.2 Captura de comandos

Este fluxo descreve o processo de captura do sistema de comandos enviados pelo usuário. O diagrama de sequência que representa este fluxo se encontra na figura 16 abaixo.

Figura 16 - Fluxo de captura de comandos



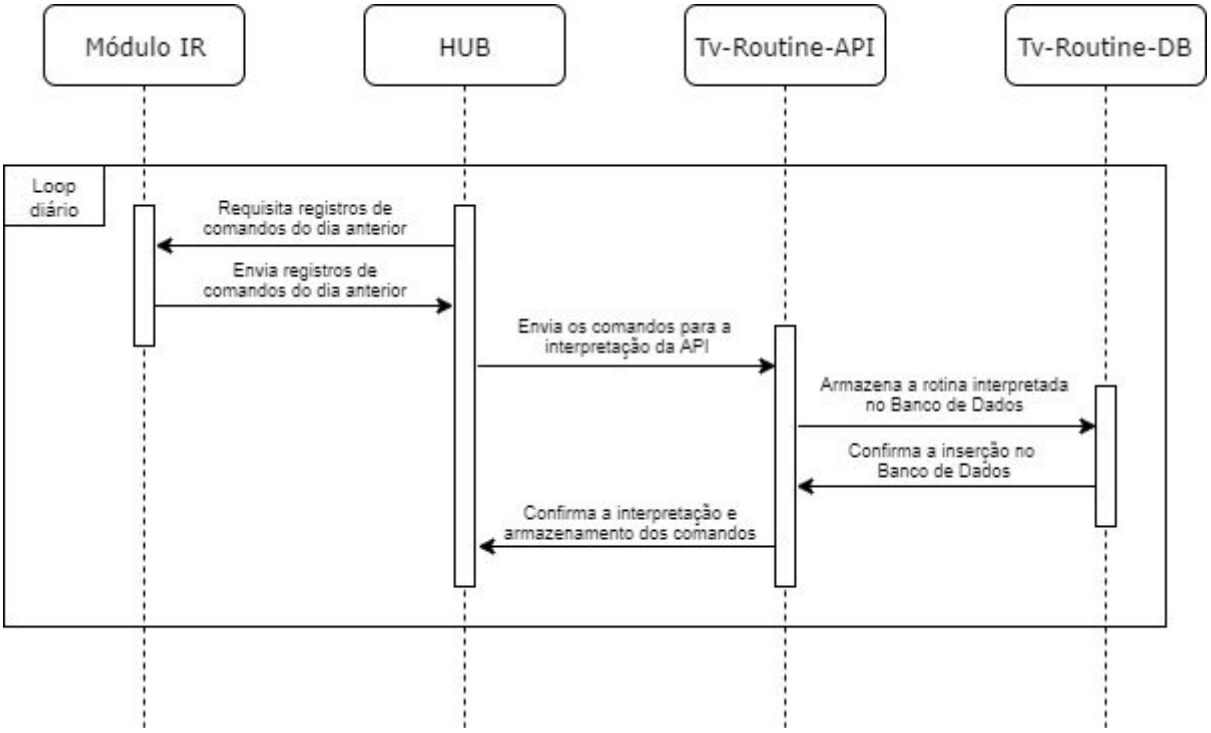
Fonte: (Própria, 2019)

Ao acionar um comando pelo controle remoto, o usuário envia a informação tanto para o receptor NET quanto para o módulo IR. Ao captar o comando, o módulo o registra em seus logs locais.

6.2.5.3 Interpretação de comandos diária

Este fluxo descreve a interpretação do sistema quanto aos comandos do usuário registrados diariamente. O diagrama de sequência que representa este fluxo se encontra na figura 17 abaixo.

Figura 17 - Fluxo de interpretação de comandos diária



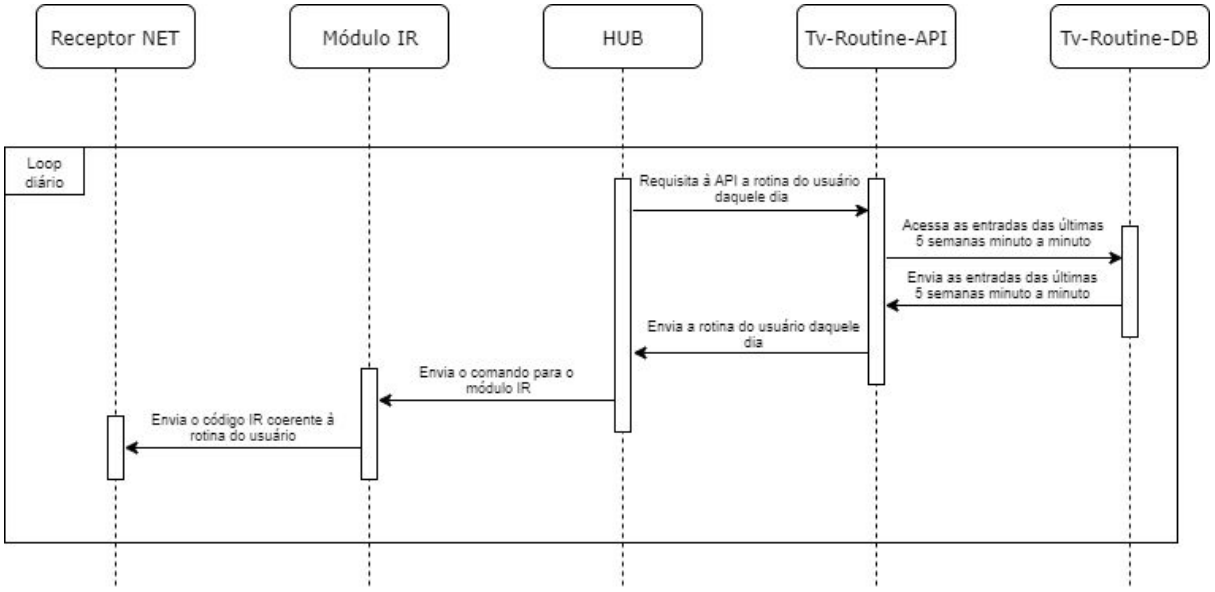
Fonte: (Própria, 2019)

Esta operação ocorre diariamente às 00h. Neste horário, o hub solicita ao módulo IR os registros de comandos do dia anterior registrados de acordo com o fluxo descrito na seção 6.2.5.2 e parte do fluxo descrito na seção 6.2.5.1. Recebidos os registros, o hub os encaminha para a API, que de fato interpreta os comandos para detectar o canal assistido, bem como seu tempo de visualização, e registra esses dados no banco.

6.2.5.4 Análise de rotina diária

Este fluxo descreve a análise de rotina diária do sistema. O diagrama de sequência que representa este fluxo se encontra na figura 18 abaixo.

Figura 18 - Fluxo de análise de rotina diária



Fonte: (Própria, 2019)

Este fluxo é programado para acontecer diariamente às 00h. Primeiramente o hub requisita à API os dados da rotina do usuário referente ao dia vigente, sendo estes gerados a partir do fluxo descrito na seção 6.2.5.3. A API, então, acessa o banco de dados para obter a informação solicitada e repassá-la ao hub. Em seguida, o hub se programa para enviar nos horários definidos os comandos de TV ao módulo IR, que de fato acionará o receptor NET para realizar as ações.

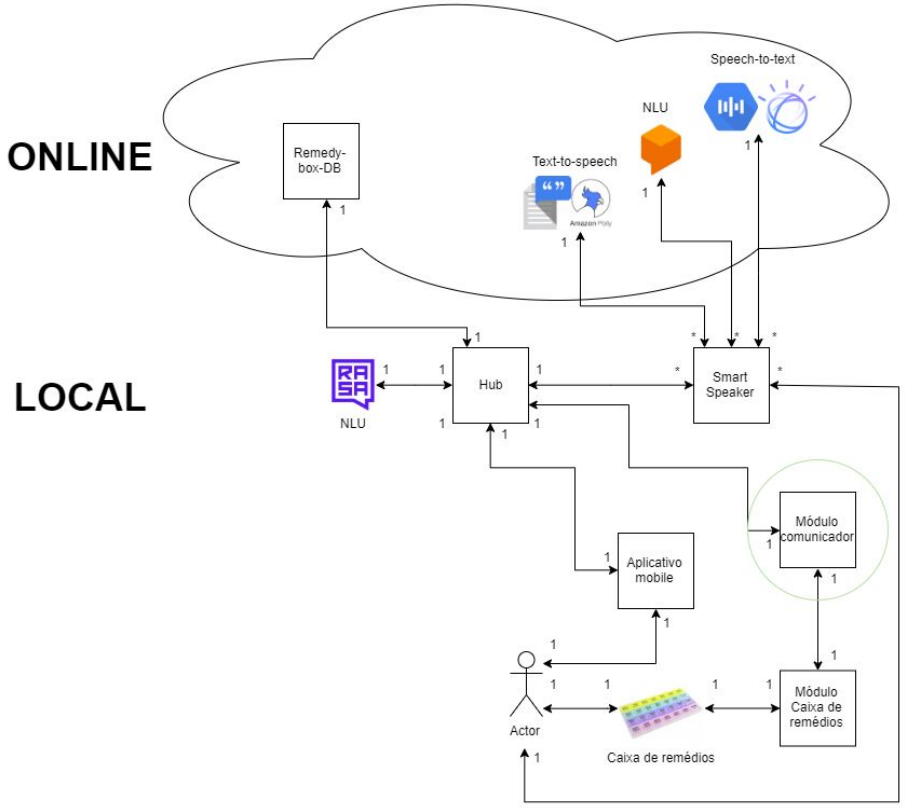
6.3 JORNADA REMÉDIO

Esta seção descreve o projeto e implementação referente à jornada de remédios. É apresentada a arquitetura da jornada, os módulos que a compõe, a caixa física criada, a API utilizada e os fluxos que fazem parte dessa jornada.

6.3.1 Arquitetura

A arquitetura referente à jornada da TV se encontra abaixo, na figura 19.

Figura 19 - Arquitetura da Jornada do Remédio



Fonte: (Própria, 2019)

Analogamente ao visto na jornada da TV, a arquitetura desta jornada se distribui entre componentes locais e online. Na nuvem se encontra o banco de dados da rotina, o serviço de NLU, o serviço de *text to speech* e o serviço de *speech to text*. Já na parte local, se encontram o *smart speaker*, o hub, o Módulo controlador dos remédios e a caixa de remédios.

6.3.2 Módulos

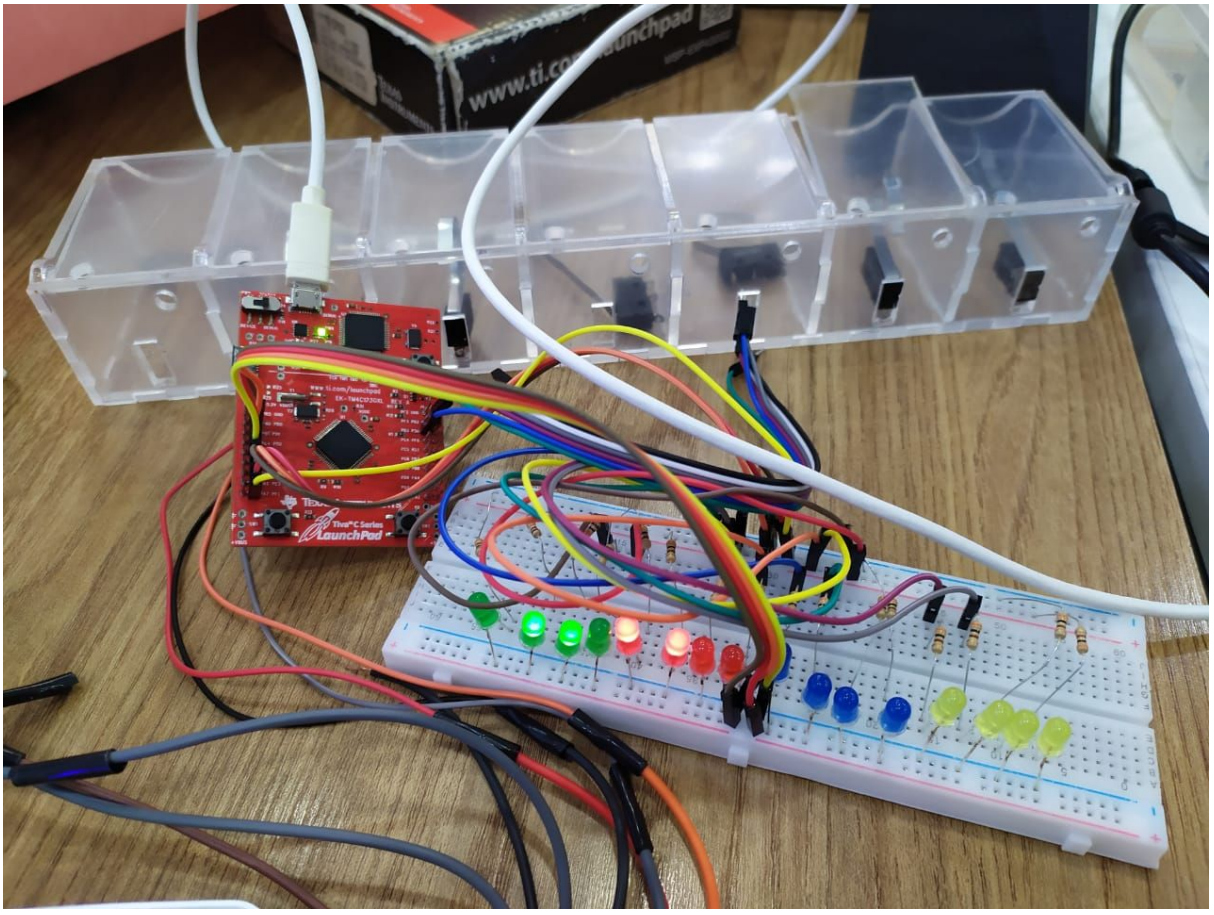
A seguir, são apresentados os módulos que compõem o sistema referente à jornada de remédios, sendo eles: o módulo controlador da caixa, responsável por acionar e captar diretamente as funcionalidades da caixa física, o módulo de comunicação, responsável por realizar a comunicação do módulo controlador com o hub - e, portanto, com o restante do sistema - bem como o hub, módulo centralizador do sistema e responsável, nesta jornada, por deter a inteligência dos fluxos e realizar a comunicação com a API e o banco de dados.

6.3.2.1 Módulo controlador da caixa

O módulo controlador da caixa é responsável por detectar a abertura dos compartimentos de remédio e de emitir alertas visuais e sonoros por meio de LEDs e um buzzer, respectivamente.

O dispositivo foi construído a partir de uma placa Tiva, descrita com maiores detalhes na seção 3.4.3. A partir desta placa, realiza-se a conexão do módulo controlador com os sensores de abertura de compartimento, os LEDs de sinalização e o *buzzer* da caixa de remédios. Todos estes componentes são conectados aos pinos de entrada/saída da placa, possibilitando realizar suas leituras ou acionamentos. A figura 20 mostra a caixa de remédio com o circuito montado.

Figura 20 - Circuito caixa de remédio



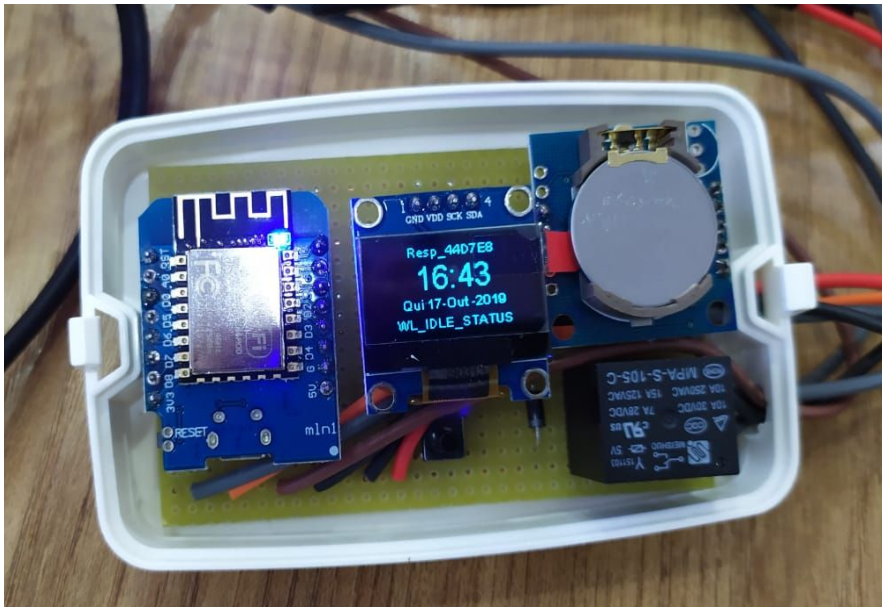
Fonte: (Própria, 2019)

6.3.2.2 Módulo de comunicação

Esse módulo é responsável por estabelecer a comunicação entre o módulo controlador da caixa com o hub integrador da casa, repassando os eventos detectados.

O módulo de comunicação é um componente existente no projeto *Hedwig*, apresentado na seção 2.1 e detalhado na seção 3.1, e foi utilizado como um módulo acelerador do projeto. Assim, sua inserção no sistema foi feita considerando-o como uma caixa preta.

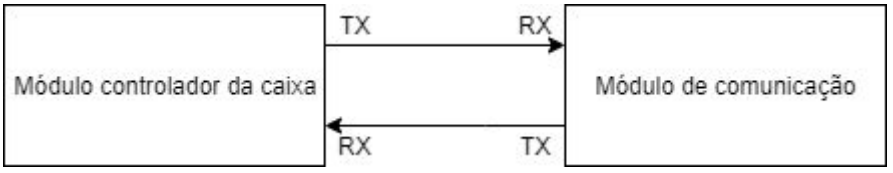
Figura 21 - Módulo de comunicação



Fonte: (Própria, 2019)

Para conectar este módulo ao módulo controlador da caixa, foi feita uma conexão utilizando os pinos seriais dos dispositivos, como mostrado na figura 22 abaixo.

Figura 22 - Conexão serial entre o módulo controlador e o de comunicação



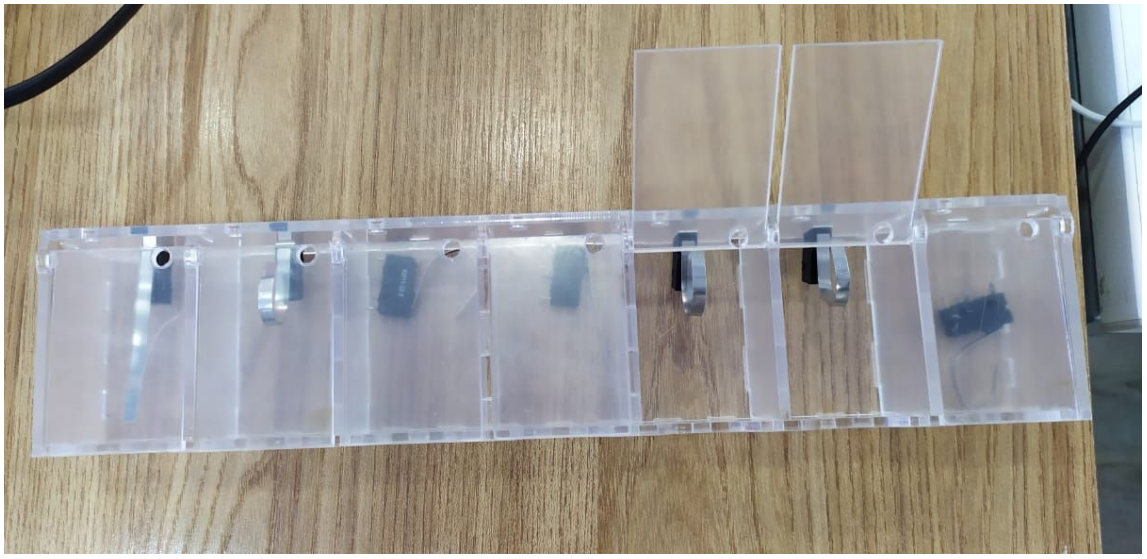
Fonte: (Própria, 2019)

Assim, é possível obter por meio do módulo de comunicação os estados de todos os sensores e LEDs da caixa, bem como ativar estes últimos. Por possuir tecnologia *wi-fi*, este módulo pode ser acessado diretamente pelo hub, provendo assim as informações necessárias para a execução dos fluxos da jornada, bem como provendo um meio de comunicação ao hub para atuar na sinalização da caixa.

6.3.3 Caixa

Para o projeto desta jornada, foi projetada uma caixa de remédios exibida na figura 23 abaixo. O projeto e confecção da componente física da caixa é de autoria de Eric Nozomi Tatsuta, também estudante da Escola Politécnica da USP e participante do programa de residência de software do Bradesco, apresentado na seção 4.3.

Figura 23 - Estrutura física da caixa de remédio



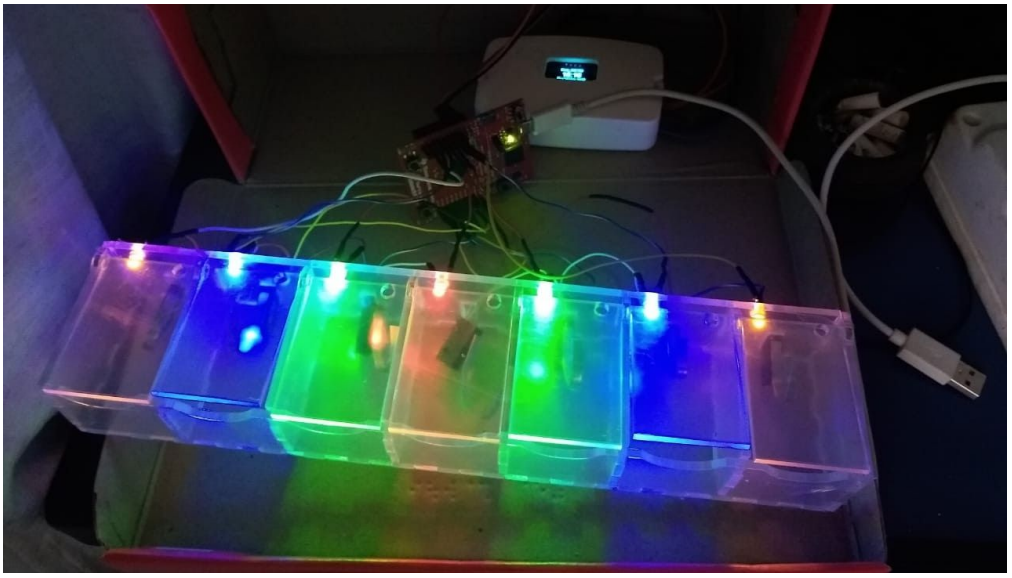
Fonte: (Própria, 2019)

Na estrutura física da caixa de remédios, são acopladas as chaves fim-de-curso e os LEDs de sinalização.

As chaves são posicionadas em cada um dos compartimentos de forma que, ao fechar a tampa, tem-se a compressão da chave até seu acionamento, detectando o fechamento. Da mesma forma, ao abrir a tampa, tem-se a descompressão da chave de forma que é detectada sua abertura.

Já os LEDs são posicionados na caixa para que haja um LED por compartimento e que ele ilumine de forma suficiente para colorir o compartimento com sua cor, assim como exibido na figura 24.

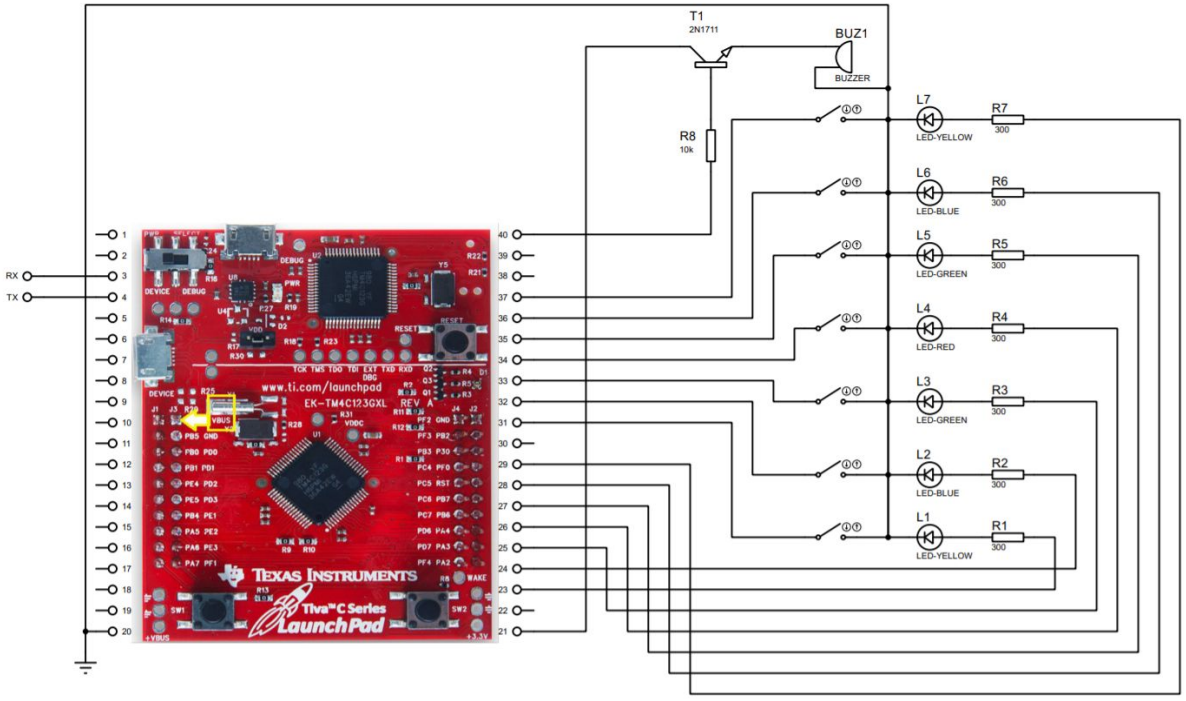
Figura 24 - Caixa de remédio em funcionamento



Fonte: (Própria, 2019)

Tanto os LEDs quanto as chaves são conectados ao módulo controlador da caixa. O circuito elétrico dessa conexão é exibido na figura 25 abaixo, criada no *software* Proteus.

Figura 25 - Circuito da caixa de remédio



Fonte: (Própria, 2019)

6.3.4 API

A API tem a função de receber do usuário a sua rotina de medicamentos, armazenando essas informações no banco de dados.

É possível enviar uma requisição para adicionar um medicamento, enviando como parâmetros o nome, dia da semana, horário, criticidade e o número do compartimento. Além disso, é possível enviar uma confirmação de consumo, enviando o horário e o número da caixa.

As entradas são salvas no banco de dados com um campo extra, o nível de alerta. O servidor executa uma tarefa periodicamente, através de cron jobs, verificando os lembretes salvos e o horário atual, atualizando o nível de alerta conforme necessário. Esse nível depende da criticidade indicada e do tempo de atraso, acionando o *buzzer* e o *smart speaker* conforme indicado na tabela 7 abaixo.

Tabela 7 - Tabela de criticidade

Criticidade	Acionamento do LED	Acionamento do Buzzer	Acionamento do *smart speaker*
0	Imediato	Não é acionado	Não é acionado
1	Imediato	Após 30 minutos	Após 60 minutos
2	Imediato	Após 15 minutos	Após 30 minutos

6.3.5 Fluxos

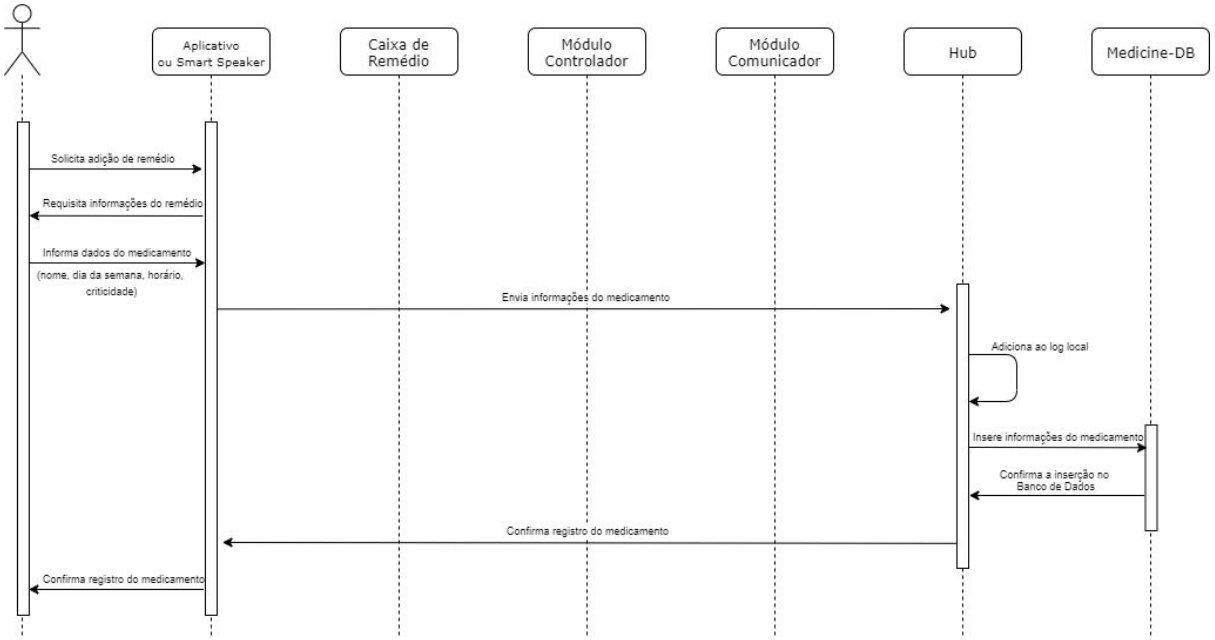
São apresentados a seguir os fluxos principais correspondentes à jornada de remédios. São eles: inserção de remédios no sistema, sinalização e alarmes, e a utilização de remédios.

6.3.5.1 Inserção de remédios no sistema

Este fluxo descreve a inserção de remédios no sistema. Isto pode ocorrer tanto via aplicativo quanto por linguagem natural via *smart speaker*. Um fluxo alternativo pode ser acionado abrindo um compartimento e inserindo um remédio.

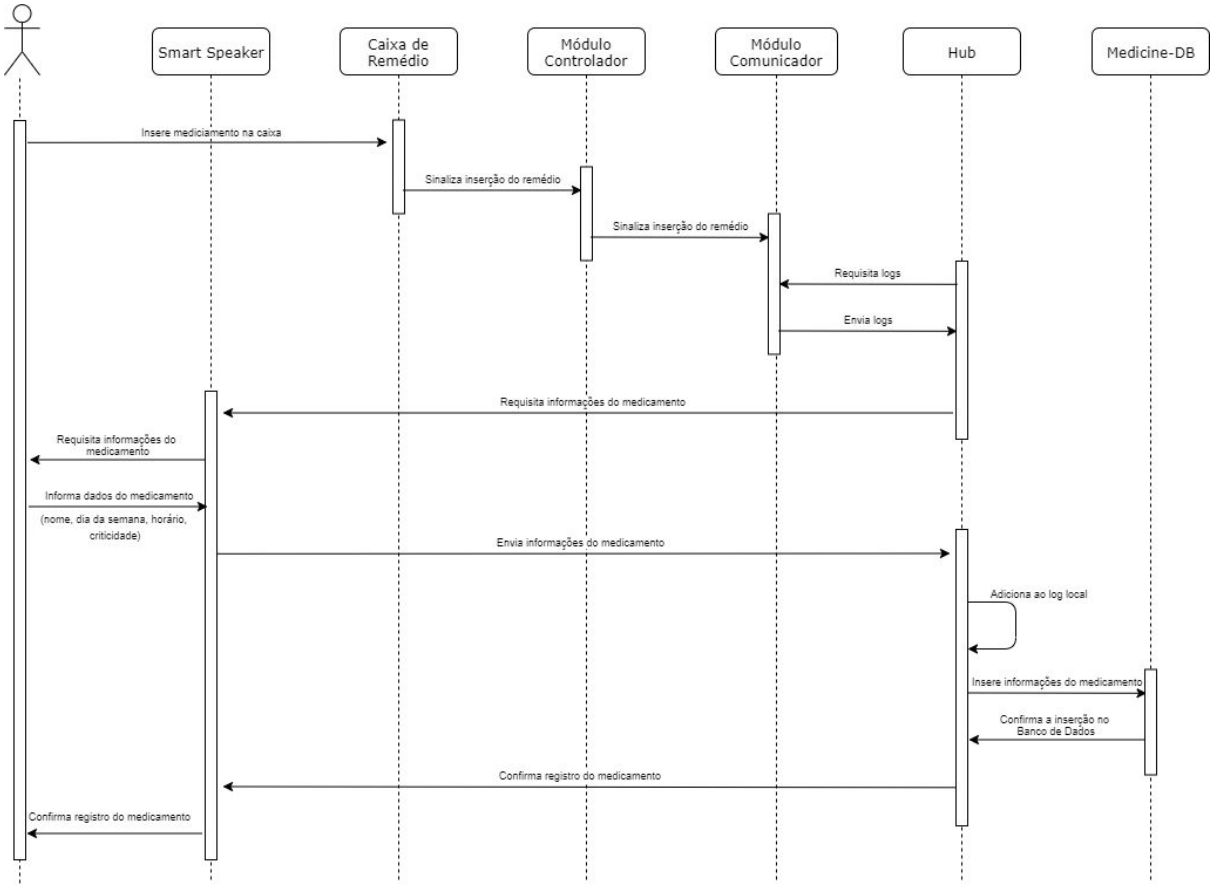
Os diagramas de sequência que representam este fluxo se encontram na figura 26 e 27 abaixo.

Figura 26 - Fluxo de inserção de remédios no sistema por *app* e *smart speaker*



Fonte: (Própria, 2019)

Figura 27 - Fluxo alternativo de inserção de remédios no sistema



Fonte: (Própria, 2019)

Para inserir um remédio no sistema via aplicativo ou *smart speaker*, o usuário deve acioná-los e informar que deseja fazer esta operação. Pelo app, isto é feito acessando a seção correspondente à inserção de remédios. Pelo *smart speaker*, basta informar isto via linguagem natural. Ambos irão, então, solicitar que o usuário informe os dados do remédio a ser inserido no sistema. Feito isso, as informações serão repassadas ao hub, que adicionará os dados ao log local de remédios. Além disso, uma redundância é feita utilizando o banco de dados.

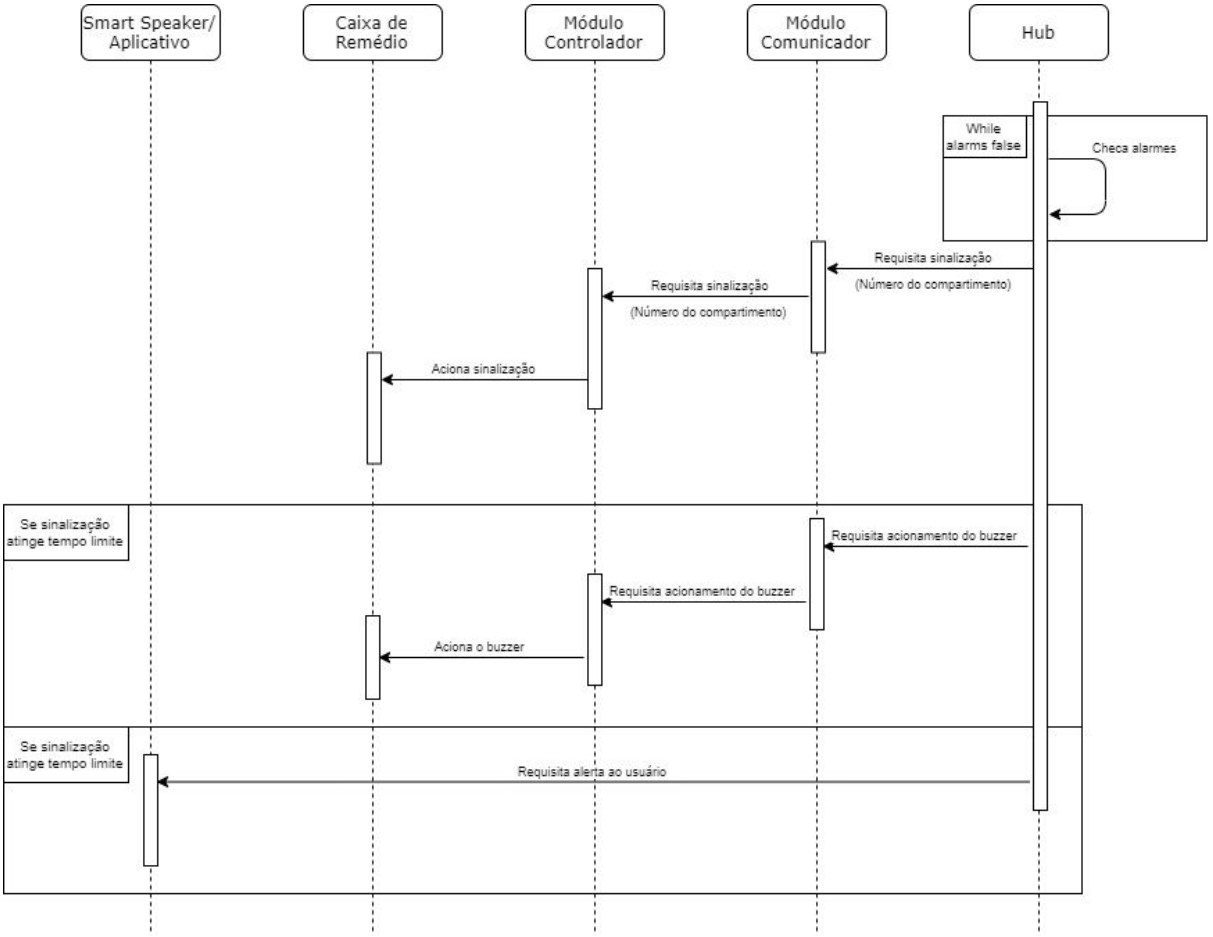
Um fluxo alternativo pode ocorrer caso o usuário abra um compartimento vazio para inserir um remédio. Neste caso, a caixa de remédios terá o sensor do compartimento acionado, informando ao módulo da caixa de remédios que isto ocorreu. Este, então, repassará a informação para o módulo de comunicação. O hub a cada 1 segundo acessa este módulo e verifica eventuais mudanças nos sensores. Desta forma, ele detecta a abertura do compartimento e aciona o *smart speaker*

para que ele solicite ao usuário as informações do remédio inserido. Isto feito, as informações são repassadas ao hub novamente, que prosseguirá o fluxo analogamente ao feito no fluxo anterior.

6.3.5.2 Sinalização e alarmes

Este fluxo descreve a sinalização e alarmes que o sistema envia ao usuário para alertá-lo de tomar seus remédios. O diagrama de sequência que representa este fluxo se encontra na figura 28 abaixo.

Figura 28 - Fluxo de sinalização e alarmes



Fonte: (Própria, 2019)

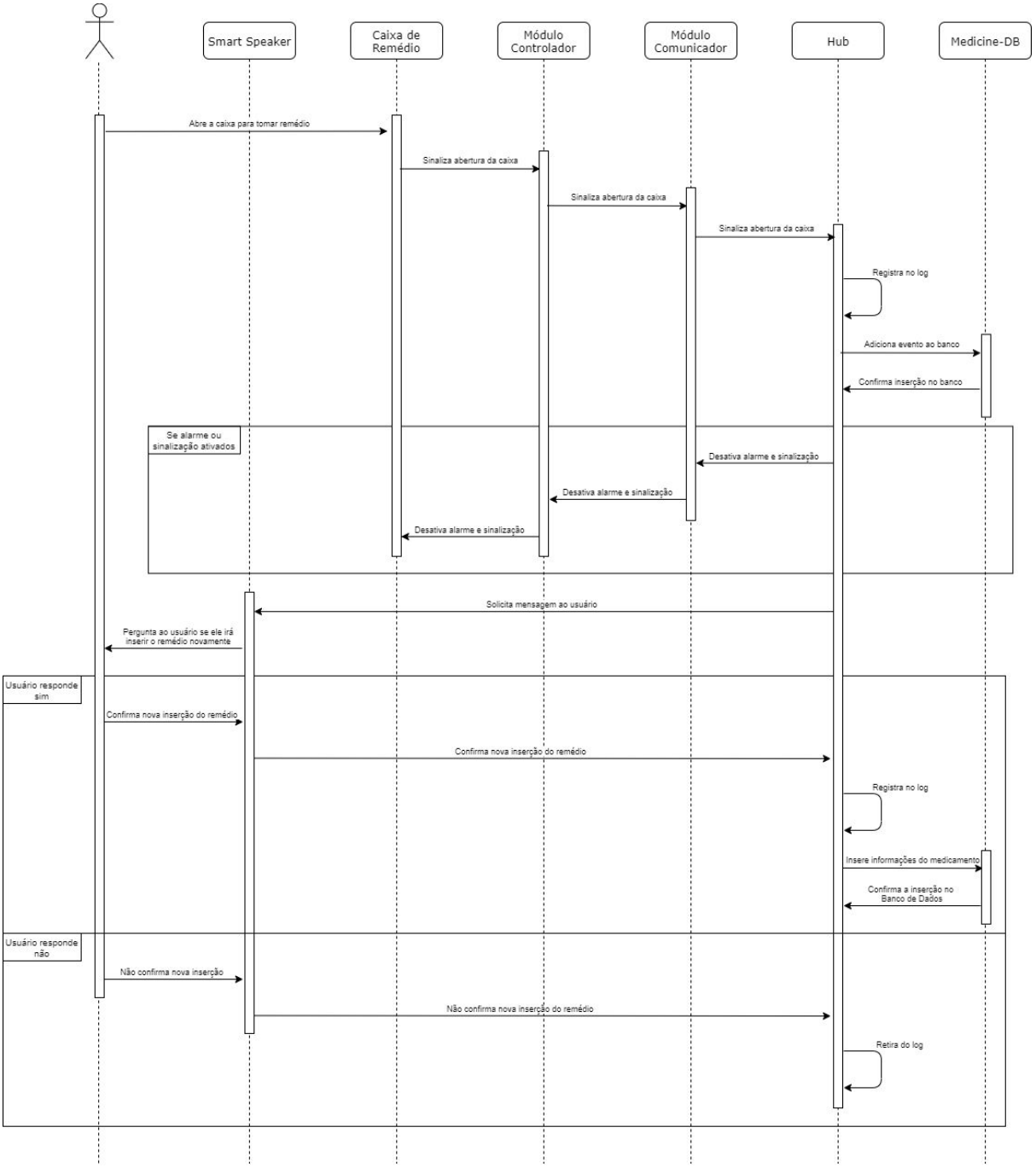
O hub constantemente se encontra em um estado de checagem de alarmes, para verificar se, naquele momento, o usuário deveria tomar algum remédio. Quando um horário deste é atingido sem que o remédio ainda tenha sido tomado, o

sistema envia ao módulo comunicador uma requisição para sinalização. Este, por sua vez, repassará o dado até o módulo controlador da caixa, que acionará a caixa para sinalizar o compartimento correspondente ao remédio. Caso o tempo limite de sinalização seja ultrapassado de acordo com a criticidade do remédio descrita na seção 6.3.4, o hub acionará o *buzzer*. Analogamente, se o tempo limite do *buzzer* for atingido, o hub acionará o *smart speaker* para alertar o usuário em linguagem natural para tomar seu remédio.

6.3.5.3 Utilização de remédios

Este fluxo descreve a utilização de remédios pelo usuário e reação do sistema a isso. O diagrama que representa este fluxo se encontra na figura 29 abaixo.

Figura 29 - Fluxo de utilização de remédios



Fonte: (Própria, 2019)

Quando o usuário abre um compartimento para tomar um remédio, esta informação é detectada pelos sensores da caixa e consequentemente obtida pelo módulo controlador. Este, então, utiliza o módulo comunicador para passar a informação até o hub. Nele, a informação é registrada nos logs locais e enviadas ao banco de dados. Caso haja alguma sinalização ou alarme ativados, o hub envia uma solicitação para desativá-los, passando pelo módulo comunicador e pelo

módulo controlador até de fato ser desativado na caixa. Na sequência, o hub envia ao *smart speaker* uma solicitação para que seja perguntado ao usuário se ele desejar adicionar uma nova dose àquele compartimento. O *smart speaker*, então, envia a resposta ao hub. Caso o usuário confirme, o registro é feito no log local e no banco de dados. Caso não confirme, ele apenas apaga do log aquele remédio.

6.4 APLICATIVO MOBILE

Esta seção descreve o aplicativo mobile desenvolvido como interface alternativa do projeto. O app tem em sua tela inicial dois botões claramente visíveis que redirecionam para as funcionalidades de TV e medicamentos, como mostrado na figura 30 abaixo e descrito nas seções 6.4.1 e 6.4.2, respectivamente.

Figura 30 - Página inicial do *app*

Fonte: (Própria, 2019)

6.4.1 Medicamentos

O fluxo do cadastro de medicamento com o app difere do fluxo com *smart speaker* por não haver participação dos componentes físicos, sendo também

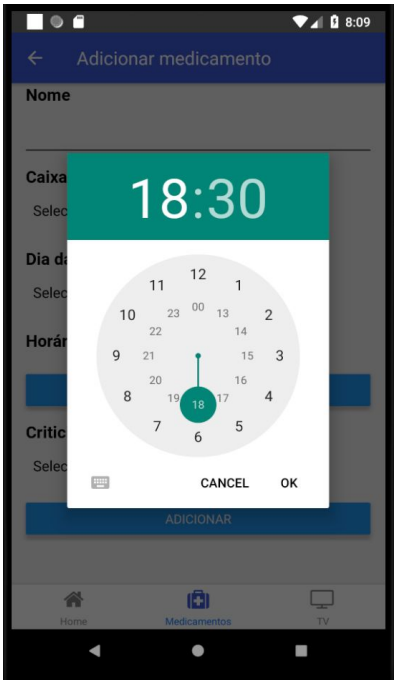
necessário informar o número da caixa. Pode servir para casos em que o sensor de abertura falhe, ou caso seja a preferência do usuário.

Após tocar no botão ou na aba de medicamentos, é exibida uma lista dos lembretes já cadastrados. Ao selecionar a opção 'Adicionar lembrete', o formulário para inserção lembretes é exibido como mostrado nas figuras 31 e 32 abaixo, iniciando o fluxo descrito em 6.3.5.1.

Figura 31 - Adição de medicamento no *app*

Fonte: (Própria, 2019)

Figura 32 - Agendamento do medicamento



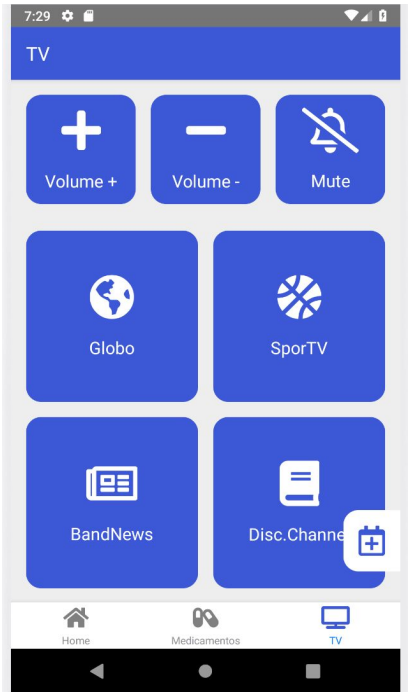
Fonte: (Própria, 2019)

6.4.2 TV

Já para a TV, o aplicativo permite o registro manual de lembretes, complementando os lembretes automatizados.

Similar aos medicamentos, o usuário pode navegar para a página de TV e, além de uma listagem simples, pode usar o botão de adição para iniciar o fluxo descrito em 6.2.5.1.

Figura 33 - Tela de comandos da TV



Fonte: (Própria, 2019)

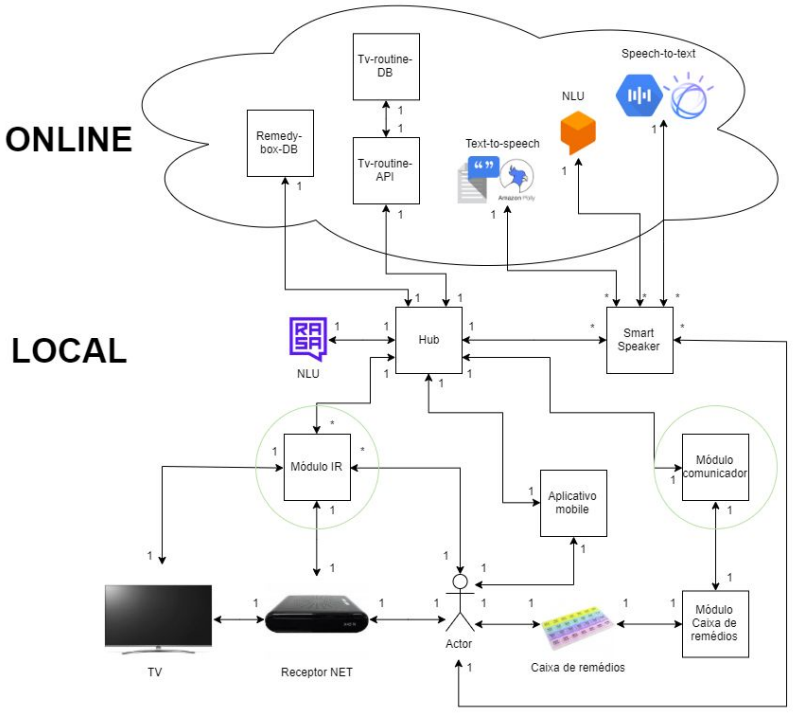
6.5 SISTEMA

Esta seção descreve o funcionamento do sistema criado como um todo, integrando o *smart speaker* e o app apresentados na seção 6.1 e 6.4 com as jornadas de *smart TV* e remédios, descritos nas seções 6.2 e 6.3, respectivamente. Assim, são descritos a arquitetura do sistema e seus fluxos que tratam de adicionar inteligência e utilidade à integração das rotinas apresentadas.

6.5.1 Arquitetura

A arquitetura geral do sistema se encontra abaixo na figura 34.

Figura 34 - Arquitetura geral do sistema



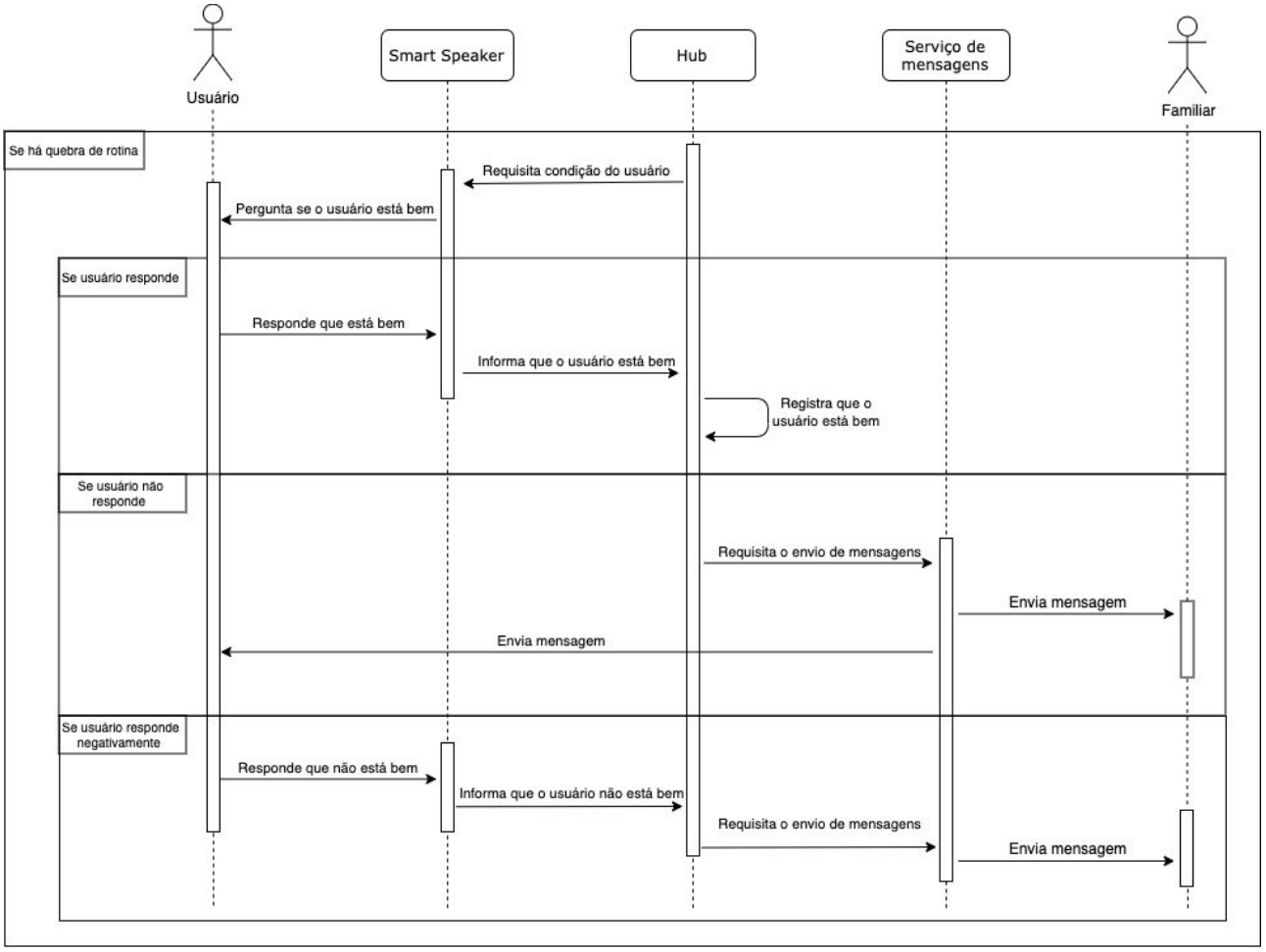
Fonte: (Própria, 2019)

O sistema como um todo também é composto por componentes locais e remotos, com o Hub centralizando o processamento de informações e gerenciando as jornadas, sendo possível acrescentar novas.

6.5.2 Fluxo

Esta seção descreve o fluxo de detecção de quebra de rotinas do sistema. O diagrama que representa este fluxo se encontra na figura 35 abaixo.

Figura 35 - Fluxo de quebra de rotina



Fonte: (Própria, 2019)

Este fluxo somente acontece quando o hub detecta alguma anomalia nas rotinas implementadas no sistema. Quando isto ocorre, o hub tenta se comunicar com o usuário utilizando o *smart speaker* para verificar se está tudo bem com ele. Caso a resposta seja sim, o *smart speaker* repassa esta informação para o hub, que registra em seu sistema que o usuário está bem e não há motivos para preocupação. Caso o usuário responda não, o *smart speaker* informa isto ao hub,

que rapidamente acionará o serviço de mensagens integrado ao sistema para enviar uma mensagem aos familiares, alertando do perigo. Caso o usuário não responda, o hub perceberá isto e acionará, analogamente à situação anterior, os familiares do usuário, bem como o próprio usuário.

7 TESTES E RESULTADOS OBTIDOS

Este seção descreve os testes e resultados obtidos após a implementação do projeto proposto. Primeiro são discutidos os testes e resultados obtidos em relação às jornadas de TV e remédio. Depois, os testes feitos para o estudo do *smart speaker*, bem como os seus resultados. Por fim, é discutido o comportamento do sistema como um todo. Para a avaliação em questão, serão analisados os comportamentos do sistema em relação aos fluxos descritos na seção 6 e comparados aos requisitos funcionais e não-funcionais descritos na seção 5.

7.1 COMPORTAMENTO DA JORNADA DA TV

A jornada da TV foi analisada segundo os fluxos presentes na seção 6.2.5, sendo eles: envio de comandos por linguagem natural e app, captura de comandos, interpretação de comandos e análise de rotina diária.

7.1.1 Envio de comandos por linguagem natural e app

Em relação ao envio de comandos por linguagem natural e app, descrito na seção 6.2.5.1, o desempenho apresentado foi satisfatório. Com o sistema ligado, ativando o *smart speaker*, foi possível trocar para um canal utilizando frases simples e usuais. Algumas frases de testes utilizadas foram:

- “Troca pra Globo.”
- “Pode colocar na Band News, por favor?”
- “Quero assistir SporTV”

Para todas as frases testadas, que remetem a uma intenção de trocar de canal e especifica um canal incluído na inteligência do projeto, o sistema de fato trocou para o canal desejado. O mesmo resultado foi obtido acessando a seção do app correspondente à troca de canais. Estes resultados satisfazem o RF1 e o RNF1, por permitir um comando de TV por app ou voz de forma acessível. A acessibilidade foi comprovada por meio de testes com usuários com pouca

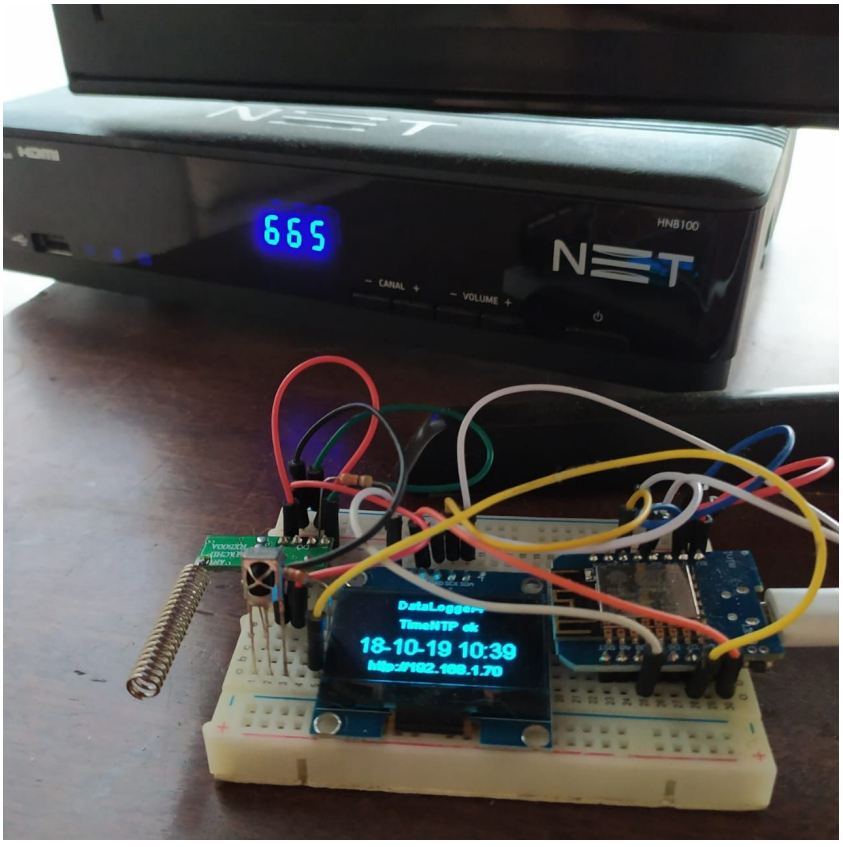
familiaridade tecnológica que conseguiram, mesmo assim, utilizar o sistema sem nenhuma complicação imediata.

7.1.2 Captura de comandos

Em relação à captura de comandos enviados pelo usuário pelo controle remoto, descrito na seção 6.2.5.2, o sistema se comportou bem, mas houve algumas constatações a respeito dos padrões de cada TV ou receptor de sinal.

Para testar a eficiência de recepção e, ainda, gerar dados de teste para análise de rotina, o módulo IR foi colocado na sala da casa de um dos integrantes do grupo, onde permaneceu colhendo dados por 3 meses. A TV presente no recinto era uma Smart TV Samsung modelo un43mu6100g e o receptor de sinal era um modelo HNB100 da Net. A figura 36 mostra o módulo IR em operação.

Figura 36 - Módulo IR da casa de um dos integrantes



Fonte: (Própria, 2019)

Observando os logs registrados pelo módulo IR, viu-se que ele de fato colhia todos os comandos do controle remoto, tanto da TV quanto do receptor de sinal da NET. Um exemplo de log é mostrado na figura 37 abaixo.

Figura 37 - *Log* de comandos IR

```
1573004673 RFRX 11803034
1573004803 RFRX 9745562
1573004821 IRRX 3782887607
1573004824 IRRX 3782887607
1573004829 IRRX 3782887607
1573004876 RFRX 11596698
1573005571 IRRX 3772782313
1573005578 IRRX 3782924327
1573005582 IRRX 3772810873
1573005587 IRRX 3782887607
1573005594 IRRX 3782912087
1573005709 IRRX 3782871287
1573005711 IRRX 3782871287
1573005711 IRRX 3782871287
1573005716 IRRX 3782897807
```

Fonte: (Própria, 2019)

Como cada comando possui um código IR distinto e que segue um padrão de fabricante, mostra-se necessário incluir estes códigos previamente no sistema, restringindo o projeto a dados já tratados. Além disso, constatou-se que alguns fabricantes incluem algumas peculiaridades nos comandos, tal como transmitir mais de uma vez o código IR ao pressionar o botão de ligar/desligar. Isso gerou desafios para tornar o sistema mais abrangente e que culminou em códigos específicos de cada fabricante dentro do projeto.

Outro desafio encontrado foi lidar com comandos que acessam menus profundos da *Smart TV*. Caso o usuário acesse o aplicativo da Netflix, por exemplo, é de extrema dificuldade monitorar sua atividade somente pelos comandos executados, uma vez que é necessário ter uma referência do menu acessado. Além disso, comandos executados diretamente nos botões da Smart TV ao invés de executados via controle remoto não foram possíveis de serem registrados devido à

arquitetura elaborada. Assim, durante o teste, orientou-se os moradores da casa a evitarem acesso a esses menus e evitarem realizar comandos diretamente na TV.

7.1.3 Interpretação de comandos diária

Em relação à interpretação de comandos diária, detalhado na seção 6.2.5.3, o sistema se comportou como esperado, mas evidenciou sua fragilidade quanto à correspondência dos dados à realidade.

De acordo com o fluxo descrito, os logs registrados no módulo IR são coletados diariamente pelo HUB, para então serem enviados à API que de fato os interpreta e os perpetua no banco de dados. Este caminho foi respeitado e executado corretamente durante os testes, ocorrendo uma vez por dia em um horário pré-determinado.

Entretanto, a interpretação dos comandos depende diretamente da qualidade de registro do módulo IR. Assim, caso não haja registro de um certo comando, a interpretação dos dados pode ser altamente prejudicada. Isto pode ocorrer em casos como desligamento do módulo IR, obstrução física do receptor, posição inadequada do módulo, além de falhas técnicas a que todos os dispositivos estão suscetíveis.

Nos testes realizados na casa de um dos integrantes, foram necessários cuidados especiais dos moradores quanto à possibilidade de ocorrência dos problemas supracitados. Respeitados estes cuidados, o sistema interpretou todos os comandos corretamente, registrando coerentemente a rotina do usuário.

Estes resultados colaboram com o RF2 e o RNF2, uma vez que ele municia o sistema com dados que serão usados na análise da rotina e não necessita da interferência direta do usuário.

7.1.4 Análise da rotina diária

Em relação à análise de rotina diária, descrita na seção 6.2.5.4, o sistema apresentou um resultado satisfatório, automatizando a rotina do usuário de acordo com os dados registrados pelo módulo IR.

A partir dos dados coletados, o sistema fez uma análise e detectou certos hábitos. Assim, o hub se programou para enviar comandos de troca de canal à TV em certos dias e horários. Estes resultados foram compatíveis com a rotina real dos usuários da casa alvo dos testes.

Assim, foram cumpridos os RF2 e o RNF2, uma vez que o sistema de fato aprendeu sobre o hábito dos usuários do teste e automatizou a rotina de forma independente, sem exigir nenhuma interferência direta.

7.2 COMPORTAMENTO DA JORNADA DO REMÉDIO

A jornada do remédio foi analisada segundo os fluxos presentes na seção 6.3.5, sendo eles: inserção de remédios no sistema, sinalização e alarmes, e utilização de remédios.

7.2.1 Inserção de remédios no sistema

A inserção de remédios no sistema, descrito na seção 6.3.5.1, apresentou resultados parcialmente satisfatórios, comportando-se como esperado na inserção via aplicativo de celular, mas apresentando alguns problemas de contexto na inserção via *smart speaker*.

Utilizando o aplicativo de celular para inserir um remédio no sistema, todo o fluxo correu bem. Foram feitos testes para inserir diferentes tipos de remédios em diferentes compartimentos da caixa de remédios e em diferentes horários e dias da semana. Para todos os casos, as alterações foram devidamente registradas no banco de dados e nos logs locais do hub.

Utilizando o *smart speaker*, porém, notou-se um grande problema a respeito da perpetuação de contextos na NLU escolhida. Diante desta dificuldade, optou-se por uma solução que driblasse a necessidade de contextos, criando uma conexão contínua entre *NLU* e o *hub*. Isto ocasionou uma demora acima do esperado na inserção do remédio via *smart speaker*.

Este resultado satisfaz o RF3 por permitir que o usuário insira remédios no sistema via app e *smart speaker*. Satisfaz parcialmente o RNF3 por mostrar que o

sistema é mais prático quando utilizado via aplicativo, apesar de que é uma opção que exige que o usuário seja levemente familiarizado com celulares, e colabora, ainda, com o RNF4, por ter de fato conseguido registrar localmente os dados dos remédios inseridos durante o teste.

7.2.2 Sinalização e alarmes

Em relação ao fluxo de sinalização e alarmes, descrito na seção 6.3.5.2, o sistema se comportou como esperado, se programando corretamente e ativando devidamente os alarmes, mesmo sem conexão direta com a internet.

Segundo o fluxo em questão, após a inserção de um remédio no sistema, é esperado que o hub se programe independentemente, incluindo em seu log previsões de sinalização e alarmes de acordo com os remédios vigentes da caixa de remédio.

Durante o teste realizado, a caixa de remédios acendeu os leds de sinalização assim que os horários estabelecidos para cada compartimento foram atingidos. Na sequência, dependendo da criticidade do remédio descrito na seção 6.3.4, o *buzzer* foi devidamente ativado, bem como o *smart speaker* acionado para alertar o usuário sobre a necessidade de tomar seu remédio. Os mesmos resultados se repetiram ao cortar a conexão do sistema com a internet.

Assim, parte do RF4 foi satisfeito, alertando o usuário corretamente nos horários estabelecidos para as doses, acionando *buzzer* e *smart speaker* quando necessário. Além disso, a independência de conexão com a internet satisfez o RNF4 que diz respeito à disponibilidade do sistema.

7.2.3 Utilização de remédios

Referente ao fluxo descrito na seção 6.3.5.3, utilização de remédios, o sistema se comportou exatamente como esperado durante os testes, desativando sinalização e alarmes e comunicando-se corretamente com o usuário para inserir um novo remédio.

Para o teste deste fluxo, inseriu-se alguns remédios na caixa, estabelecendo devidamente seus dados. Ao serem atingidos os horários das doses, o respectivo compartimento foi aberto, sinalizando que o remédio foi tomado.

O sistema comportou-se bem, desligando sinalizações e alarmes acionados quando o compartimento foi aberto. Após isso, o speaker de fato perguntou ao usuário se gostaria de incluir um outro remédio no sistema. Respondendo sim e mantendo as mesmas informações do remédio, o sistema perpetuou os dados no log local. Respondendo sim e incluindo um novo remédio, o sistema substituiu no log local os antigos dados pelos novos dados. Respondendo não, o sistema apagou os logs locais. Em todos os casos, foi inserida no banco de dados a informação de que o remédio foi devidamente tomado naquele horário específico.

Assim, este resultado cumpre o RF4 por desativar corretamente os alertas e colabora com o RNF3, já que permite ao usuário inserir um novo remédio ou repetir a dose de forma acessível e intuitiva. A acessibilidade desta jornada foi testada com um possível usuário e comprovada pelo fato de o usuário ter conseguido utilizar o sistema sem nenhuma orientação e sem exigência de conhecimentos tecnológicos prévios.

7.3 ESTUDO DE SMART SPEAKER OPEN SOURCE

Para este trabalho, foi desenvolvido um estudo referente ao *smart speaker Open Source,* que tem por objetivo propor uma arquitetura aberta de *smart speaker* a partir do conceito de SOA - *service oriented architecture*[22], bem como apresentar os testes comparativos qualitativos e quantitativos de cada um dos três serviços básicos empregados para a escolha otimizada dos serviços necessários para a construção de um *smart speaker Open Source*. Em relação aos serviços de transcrição de fala, foram analisadas as ferramentas *Google Speech to Text*[16] e *Watson Speech to text*[17]. Já para o serviço de entendimento de linguagem natural, foi analisado o *Dialog Flow*[18], do *Google,* e o projeto open source *Rasa*[19].

7.3.1 Teste de serviços existentes

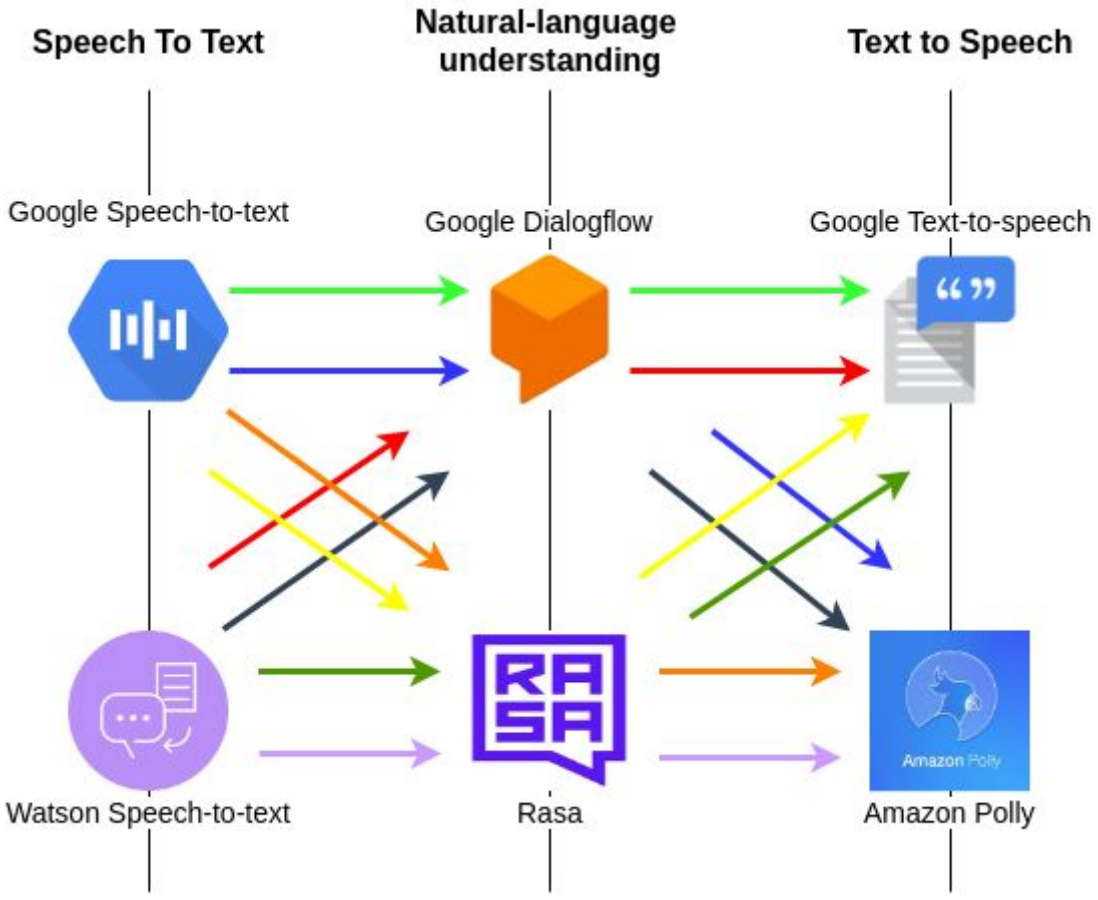
Para testar a eficiência de cada um dos três serviços utilizados no *speaker*, foi feita uma abordagem que combina aspectos qualitativos e quantitativos.

Para a abordagem qualitativa, considerou-se primeiramente a taxa de acerto do serviço de *speech to text*. Assim foram feitas quatro baterias de testes, cada uma com uma entrada diferente de áudio. Posteriormente foi analisado o quão bem o serviço foi capaz de transcrever a fala em comparação ao esperado.

Para a abordagem quantitativa, foi medido o tempo de resposta de cada um dos serviços e feita uma análise comparativa entre os mesmos.

Em ambos os testes, foi utilizada uma API construída que possibilita escolher o serviço usado em cada um dos processos, tendo as possíveis combinações exibidas na figura 38.

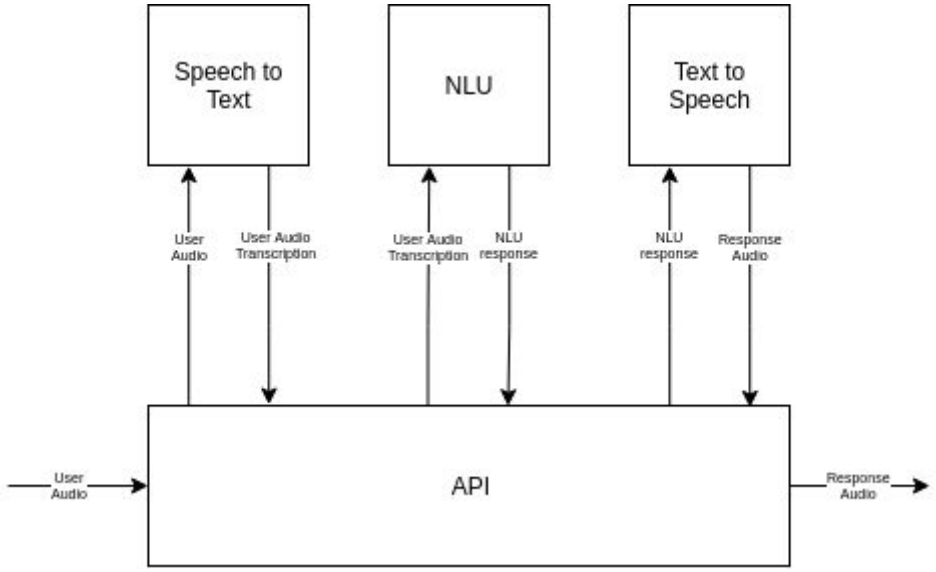
Figura 38 - Diagrama dos possíveis serviços testados



Fonte: (Própria, 2019)

A arquitetura da API e sua forma de dialogar com os serviços disponíveis é mostrado na figura 39 abaixo.

Figura 39 - Arquitetura da API com os serviços básicos



Fonte: (Própria, 2019)

A API foi executada localmente em um notebook Dell Inspiron 15 série 7000, 16GB de RAM, processador 7ª geração Corei7 com sistema operacional Windows 10 e conexão via cabo ethernet a uma internet de 60mb.

7.3.1.1 Abordagem Qualitativa

Para a abordagem qualitativa, analisou-se a transcrição feita pelo serviço de speech-to-text e áudio fornecido pelo serviço de text-to-speech. Já para a abordagem quantitativa, a aplicação foi alterada para produzir logs contendo o tempo levado por cada serviço.

No primeiro teste, foi utilizada uma fala retirada de um vídeo do youtube da Monja Coen sobre budismo. Neste áudio, a fala da monja ocorre em alto e bom tom, sem interferências externas e com muito pouco ruído.

No segundo teste, o áudio ficou por conta de um pequeno trecho do poema de Chico Buarque lido pelo próprio. A gravação contém uma música de fundo e a

fala do artista é rápida, o que pode simular bem situações possíveis que o speaker enfrentaria.

No terceiro teste, foi usada dublagem brasileira de um trecho do famoso discurso de Charles Chaplin no filme “O Grande Ditador”. Nesta versão, a voz do personagem vem acompanhada de certo eco e um ruído advindo do pouco recurso tecnológico existente na época.

Por fim, no último teste foi utilizado um trecho do discurso motivacional do jogador Rogério Ceni para o time do São Paulo antes de um importante jogo da equipe em 2013. Na gravação, é possível entender o discurso do jogador mas é necessário certo esforço, devido ao baixo volume da fala e a grande quantidade de ruído gerado pelo ambiente da gravação - o vestiário de um estádio de futebol.

Dessa forma, os testes variaram dois dos mais importantes parâmetros a se considerar nos testes dos serviços em questão: a clareza de uma fala e o ruído do ambiente.

A tabela 8 abaixo mostra os áudios utilizados e suas respectivas características quanto à fala e ao ambiente.

Tabela 8 - Áudios utilizados no teste qualitativo e suas características

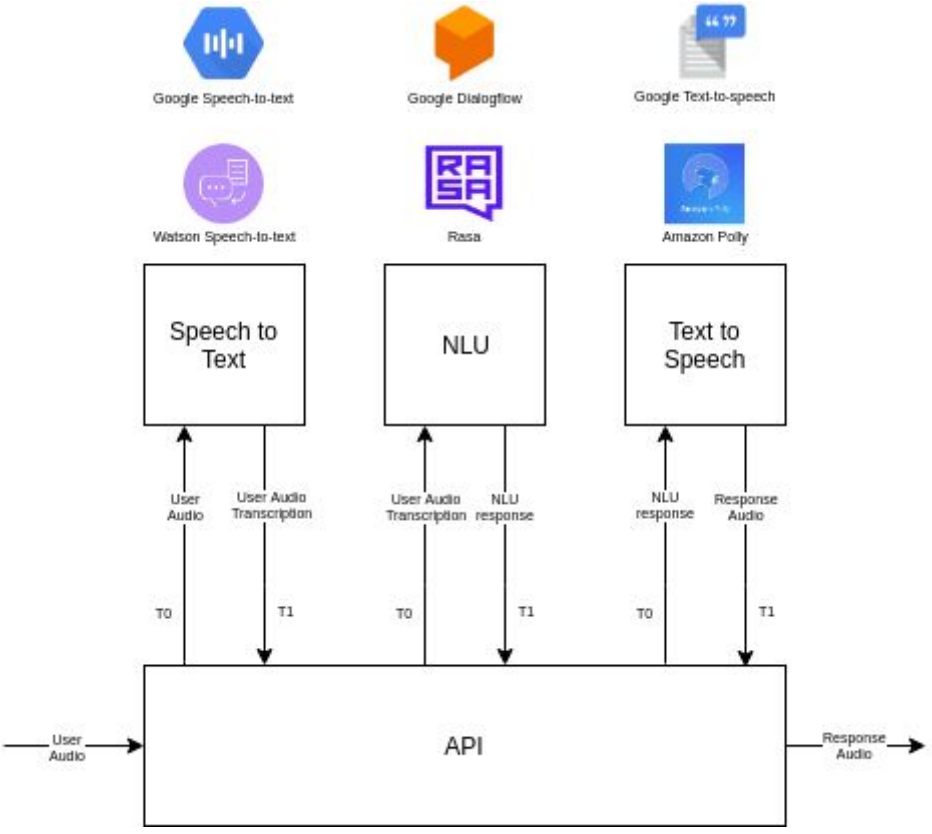
Teste	Conteúdo	Característica da fala	Ruído do ambiente
1	Fala da Monja Coen sobre meditação [23]	Muito clara	Quase inexistente
2	Leitura de um poema do Chico Buarque [24]	Clara	Música tocada durante a fala
3	Fala de Charlie Chaplin em “O Grande Ditador” [25]	Clara mas com eco e gravação de pouca qualidade	Pouco ruído
4	Discurso de Rogério Ceni [26]	Pouco clara e com gravação de pouca qualidade	Muito ruído

7.3.1.2 Abordagem quantitativa

Para a abordagem quantitativa, inseriu-se no código da API marcações de tempo que possibilitaram medir o intervalo entre o acionamento de um serviço e sua resposta. Os resultados foram registrados em um arquivo de log no formato CSV e, posteriormente, concentrados em uma planilha para serem visualmente mais explicativos. Na figura 40, esses tempos são representados por “t0” e “t1”, presentes nas setas de interação entre a API e os serviços.

Para este teste, não foi relevante a combinação de serviços utilizada, mas sim o desempenho individual de cada um deles.

Figura 40 - Arquitetura da API com os serviços básicos e tempos

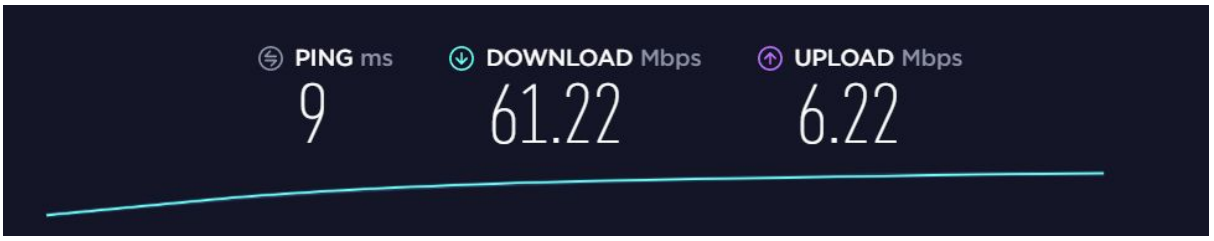


Fonte: (Própria, 2019)

Para este teste, levou-se em consideração a necessidade de uma conexão de internet estável, que pouco ou em nada influenciasse os resultados. Assim, foi utilizado um cabo de ethernet diretamente conectado no computador, além de uma conexão de 60mb. Durante todo o teste, a qualidade da conexão foi medida

utilizando-se o site SpeedTest[27], como mostra a figura 41 abaixo. Confirmou-se de fato muito pouca variação na conexão durante o teste.

Figura 41 - Conexão segundo o *SpeedTest* durante os testes



Fonte: (SPEEDTEST, 2019)

Para garantir que todos os parâmetros que poderiam influenciar o resultado fossem variados, optou-se por fazer testes com os 4 áudios descritos na tabela 8. Assim, para cada um destes, foram feitas 50 chamadas aos serviços. A partir do tempo medido em cada chamada, obteve-se uma média de tempo gasto no acionamento de cada um dos serviços em cada um dos 4 casos analisados. Por fim, foi feita uma média entre os casos e obtido um tempo médio gasto para cada um dos serviços. Dessa forma, o resultado obtido se aproxima mais do desempenho geral sem grande interferência de casos isolados.

7.3.2 Resultados dos serviços existentes

Os resultados obtidos após realizar os testes de desempenho do *smart speaker* para diferentes serviços são apresentados nesta seção, diferindo as abordagens qualitativas e quantitativas nas seções 7.3.2.1 e 7.3.2.2, respectivamente. O comportamento do *smart speaker* em relação às jornadas de TV e remédio são explicadas nos tópicos 7.1 e 7.2, uma vez que é necessário contextualização perante a jornada.

7.3.2.1 Resultado da abordagem qualitativa

Para o primeiro teste em relação aos serviços de *speech to text*, os resultados obtidos estão dispostos na tabela 9 abaixo. A primeira coluna exibe a

transcrição correta do áudio original, a segunda, a transcrição obtida pelo serviço do Google e, por fim, a terceira coluna exibe a transcrição obtida pelo serviço da IBM.

Tabela 9 - Resultados das transcrições da Google e IBM

Texto Original	Transcrição - Google Speech to Text	Transcrição - Watson
“Quando a gente fala sobre Budismo, Buda significa aquele que acordou, que despertou, aquele que percebe a realidade assim como é, de si mesmo e de tudo à sua volta.”	“a gente fala sobre o Budismo Buda significa aquele que acordou que despertou aquele que percebe a realidade assim como é de si mesmo e de tudo à sua volta”	“montante fala sobre budismo buda significa aquele que acordou que despertou aquele que percebe a realidade assim como é de se mesmo e de tudo a sua volta”
“Eu era um jovem louro e saudável quando adentrei a baía de Guanabara, errei pelas ruas do Rio de Janeiro e conheci Teresa. Ao ouvir cantar Teresa, caí de amores pelo seu idioma, e após três meses embatucado, senti que tinha a história do alemão na ponta dos dedos.”	“eu era um jovem loiro e saudável por dentro da Baía de Guanabara errei pelas ruas do Rio de Janeiro e conheci Teresa a ouvir cantar Teresa cair de amores pelo seu idioma e após três vezes batucado City que território alemão na ponta dos dedos”	“eu era jovem lore saudável quando deveria baía de guanabara hoje pelas ruas do rio de janeiro onze tereza”
“Soldados, não se entreguem a esses brutos. Homens que desprezam vocês, escravizam vocês, governam suas vidas, dizem o que devem fazer, o que pensar e o que sentir, que conduzem vocês, ditam sua comida, tratam vocês como gado e como buchos de seus canhões! Não se dediquem a esses homens sobrenaturais! Homens-máquina, com máquinas no cérebro e máquinas no coração! Vocês não são máquinas! Vocês não são gado! Vocês são homens!”	“homens que desprezam vocês não te aviso vocês compraram Duas Vidas o que deve fazer o quê parceiro que sentido e conduzem você traz as máquinas do coração Conceição sammartino”	“contracheque propor <comerc> para convocando o câmbio não protege contra um clérigo tarifa técnica para que com o gol.”

“Eu sei que vocês acreditam em Deus, não acreditam todos vocês? Que isso aqui vire um inferno lá dentro. Que seja um inferno hoje aqui dentro. Na hora que vocês subirem dentro do campo e verem aqueles cara ali em cima, olha pra cima, olha onde vocês tão, olha o que vocês tão usando, olha a oportunidade que Deus deu na vida pra cada um de vocês, como time, como ser humano, como homem, como pai de família (bora mano), olha a oportunidade que Deus tá dando pra cada um de vocês (isso aí galera) ESCREVER HISTÓRIA, ESCREVER HISTÓRIA”	“onde vocês estão vocês estão usando agora que tem o teu na vida para cada um de vocês como o ser humano o homem como vai você”	“com o”

Após obter os resultados dos 4 testes apresentados na tabela acima, a conclusão que se obtém é que o serviço de *speech to text* da *Google* foi superior ao da IBM em todos os casos.

No primeiro caso, em que se utilizou o áudio da monja Coen de fala muito clara e ruído quase inexistente, ambos os serviços apresentaram resultados muito satisfatórios, aproximando-se muito do texto original. As discrepâncias ficaram por conta de pontuações e alguns erros esporádicos. Ainda assim, o resultado do Google se aproximou mais do esperado, ainda que de forma muito sutil.

No segundo caso, já é possível observar o fator que mais distingue os dois serviços analisados: o resultado obtido pelo *Watson* deixou de transcrever muitas partes do áudio. A fala de Chico Buarque era clara mas possuía uma música de fundo que gerava certo ruído, fato este que provavelmente foi um dos empecilhos encontrados pelo *Watson* e que o levou a fazer uma transcrição incompleta. Por outro lado, o serviço do *Google* completou a transcrição, apresentando alguns erros, muitos deles significativos e que poderiam prejudicar o entendimento de alguns pontos da fala Assim, pode-se dizer que sua transcrição transmite a mensagem principal do áudio.

No terceiro caso, constituído por uma fala de Charles Chaplin clara mas com eco, gravação de pouca qualidade e alguns ruídos, ambos os resultados foram muito ruins e não conseguiram sequer transmitir a ideia central do áudio. A transcrição do *Google* ainda foi capaz de reconhecer algumas palavras, mas ignorou muitas e errou outras. Já a transcrição da IBM praticamente não acertou nenhuma palavra do áudio, além de apresentar um resultado estranho no meio do texto - <comerc>.

Por fim, o último caso tinha apenas a intenção de testar os limites dos serviços, usando uma fala de Rogério Ceni pouco clara, com gravação de pouca qualidade e muito ruído, um dos piores cenários possíveis para um serviço de transcrição de áudio. Como esperado, ambos os resultados foram ruins e não conseguiram transmitir, novamente, a ideia central do áudio. Entretanto, o Google Speech-to-text não apresentou uma queda de rendimento muito significativa do terceiro para o quarto caso. A transcrição apresentou corretamente algumas palavras do texto original, ainda que no contexto geral o resultado não tenha sido bom.

7.3.2.2 Resultado da abordagem quantitativa

Os resultados do teste com abordagem quantitativa são exibidos na tabela 10 abaixo.

Tabela 10 - Tempo médio para cada teste e tempo médio final

Áudio	STT Google	STT Watson	NLU Dialog Flow	NLU Rasa	TTS Google	TTS Polly
Monja Coen	6819.13	6908.69	1062.88	17.38	1447.50	930.02
Chico Buarque	11231.49	9724.42	1142.06	17.05	1541.52	919.11
Charles Chaplin	30545.66	17118.36	1057.99	19.58	1313.49	889.09
Rogério Ceni	20502.13	12695.51	1088.37	17.09	1433.75	691.28
Média	17274.60	11611.74	1087.83	17.77	1434.06	857.38

Como se pode verificar na tabela 10, os resultados para diferentes condições de áudio foram distintos para os três serviços.

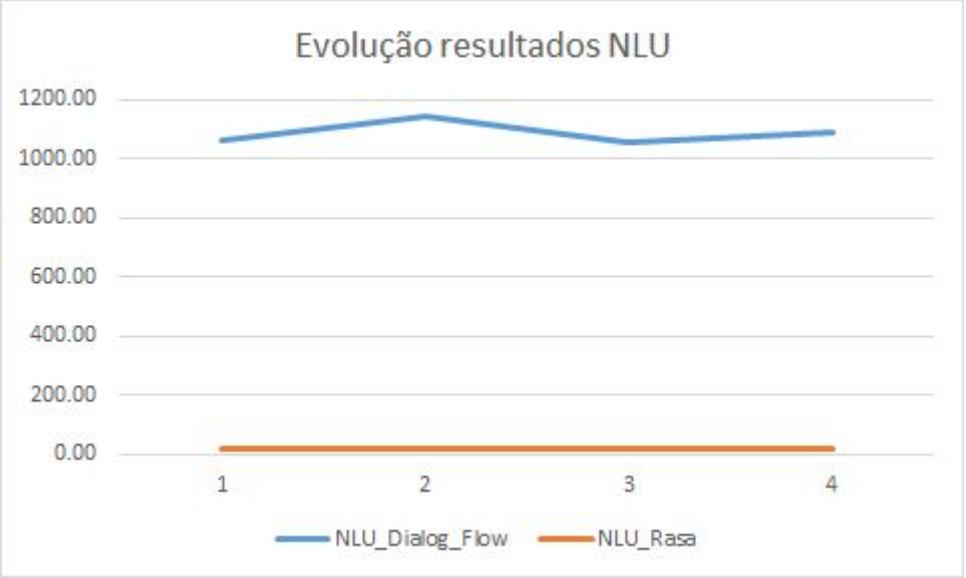
No caso do *speech to text*, quando o áudio era claro, tanto o serviço da Google quanto o da IBM apresentaram desempenho semelhante. Todavia, conforme a qualidade do áudio dos testes foi diminuindo, a diferença de desempenho foi se acentuando até chegar a uma proporção próxima a 2:1, quando a diferença voltou a cair. Esta evolução é mostrada abaixo, na figura 42.

Figura 42 - Evolução dos resultados de Speech-to-text

Fonte: (Própria, 2019)

Já para o caso da NLU, independentemente do teste, os tempos de processamento não variaram muito e, portanto, também não foi significativa a diferença de desempenho entre os serviços, como se pode verificar na figura 43.

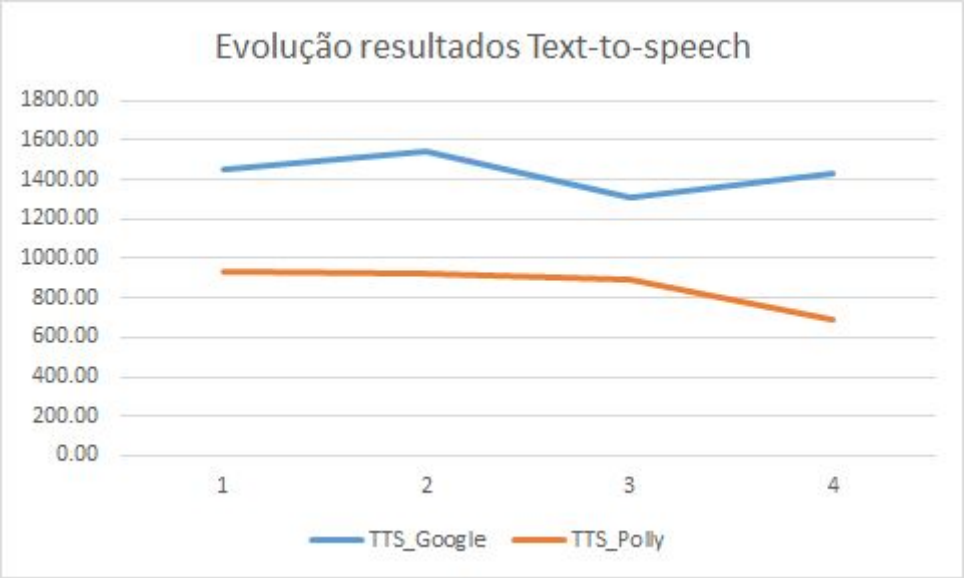
Figura 43 - Evolução dos resultados de NLU



Fonte: (Própria, 2019)

Por fim, para o caso do *text to speech*, a diferença de desempenho entre os dois serviços teve uma constante variação. Diferentemente do caso de *speech to text,* para estes serviços, não foi possível identificar uma relação clara entre o caso analisado e o tempo de resposta, levando a concluir que as variações são provenientes somente de oscilações no tempo de resposta dos serviços.

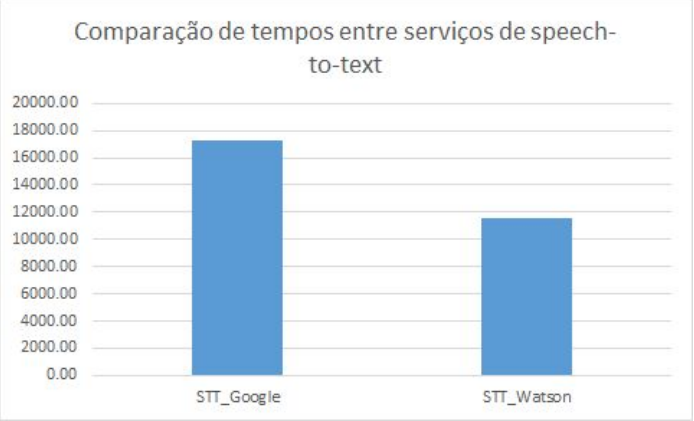
Figura 44 - Evolução dos resultados de Text-to-speech



Fonte: (Própria, 2019)

Abaixo são exibidas as figuras contendo os gráficos comparativos entre as médias finais dos tempos de resposta dos serviços analisados neste estudo.

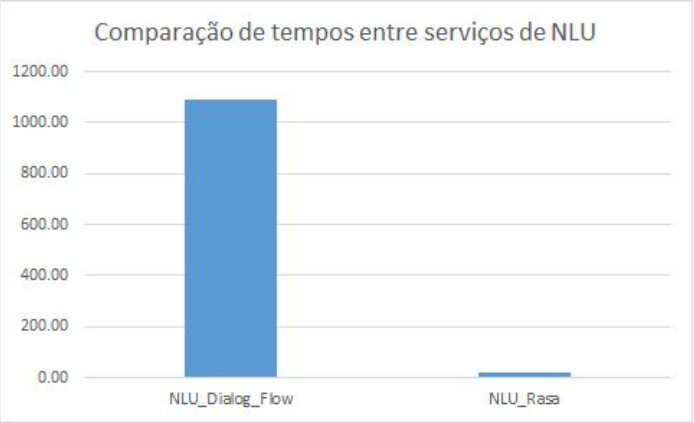
Figura 45 - Comparação de tempos entre serviços de Speech-to-text



Fonte: (Própria, 2019)

Como pode-se observar pelo gráfico exibido na figura 45 acima, ao comparar os serviços de speech-to-text, o serviço oferecido pela IBM apresentou melhor desempenho em detrimento do serviço oferecido pela Google. A diferença do tempo médio de resposta entre ambos foi de 7.021ms, o que representa uma resposta aproximadamente 68% maior do Google speech-to-text em relação ao IBM Watson. Como visto na figura 45, a diferença mais acentuada entre os serviços foi identificada quando o áudio possuía muitos ruídos e pouca clareza na voz - áudio do Charles Chaplin. Deve-se levar em consideração, portanto, que esta diferença de tempo é subjetiva, uma vez que, no caso de maior discrepância, o serviço da IBM apresentou uma transcrição muito inferior ao serviço da Google.

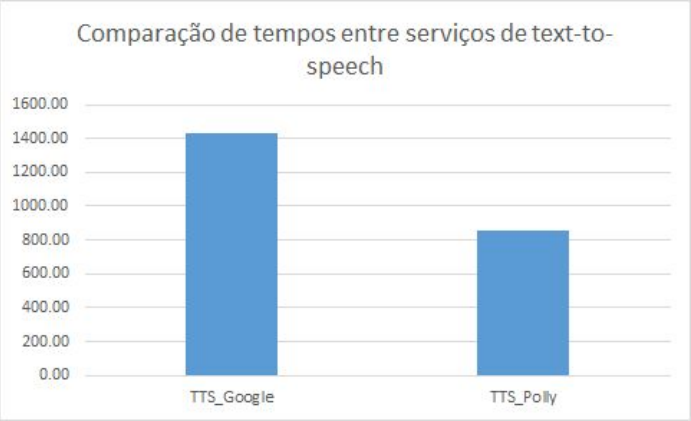
Figura 46 - Comparação de tempos entre serviços de NLU



Fonte: (Própria, 2019)

A figura 46 aponta o resultado mais significativo na comparação de tempo de resposta entre os serviços analisados: o tempo de resposta do Dialog Flow foi acima de 60 vezes mais lenta do que o do Rasa, apresentando uma diferença de 1103ms entre ambos. Isso se deve ao fato de o Rasa ser um serviço que roda localmente, sem exigir um tráfego de dados via internet, o que também fornece maior disponibilidade em relação a serviços remotos.

Figura 47 - Comparação de tempos entre serviços de Text-to-speech



Fonte: (Própria, 2019)

Já para os desempenhos dos serviços de *text to speech*, como se pode verificar na figura 47, a diferença de tempo de resposta foi favorável ao *Amazon Polly*, apresentando uma resposta aproximadamente 598ms mais rápida que a do serviço do *Google*.

8 CONSIDERAÇÕES FINAIS

O capítulo final deste trabalho apresenta as conclusões obtidas do projeto de formatura, sua relevância na área e apresenta possíveis trabalhos futuros.

8.1 CONCLUSÕES DO PROJETO DE FORMATURA

Considerando os objetivos iniciais deste projeto e após analisar os resultados obtidos, pode-se concluir que o objetivo principal deste trabalho foi atingido. O sistema desenvolvido atuou bem nos testes das jornadas de TV e remédio propostos, mostrando-se uma opção viável para auxiliar na independência de um idoso. Em relação à jornada da TV, o sistema foi capaz de auxiliar o usuário nas ações básicas e ainda automatizar sua rotina, facilitando o acesso a um entretenimento. No tocante à jornada do remédio, o sistema foi capaz de prover uma forma organizada e automatizada do usuário ingerir seus remédios, podendo ser importante auxílio em uma rotina relacionada à saúde. Além disso, as duas interfaces criadas comportaram-se muito bem, permitindo ao usuário tanto utilizar uma forma mais rápida e eficiente como o aplicativo, quanto utilizar o *smart speaker* para se comunicar com o sistema utilizando linguagem natural.

8.2 CONTRIBUIÇÕES

Este trabalho tem como maior contribuição confirmar a viabilidade técnica de um sistema de *smart home* que seja adequado para auxiliar no dia a dia de um idoso que queira manter sua independência. Também, o fato deste trabalho ter sido desenvolvido em parceria com o Bradesco por meio de seu programa de residência, faz com que este projeto seja parte de algo maior, com grande potencial de crescimento. Futuramente, alunos que queiram desenvolver seu trabalho de conclusão de curso dentro desta área podem utilizar este sistema ou parte dele, assim como foi feito com o projeto Hedwig do Engenheiro Victor Hayashi, base arquitetural deste trabalho. Além disso, a arquitetura *open source* estudada e construída para o *smart speaker* se mostra um estudo relevante no ramo de

assistentes pessoais, que ainda não conta com um projeto similar presente no mercado.

8.3 TRABALHOS FUTUROS

Posto que este trabalho de conclusão de curso compõe um sistema maior de *smart home* e se apresenta como uma confirmação da viabilidade da arquitetura, muitas são as opções para melhorá-lo e expandi-lo. Trabalhos futuros podem concentrar-se tanto na evolução das jornadas da TV e remédio, de forma a aprimorá-las, assim como na criação de novas jornadas e novas funcionalidades do sistema como um todo.

Na jornada da TV, pode-se citar primeiramente a expansão dos comandos possíveis para controle da TV, uma vez que no projeto elaborado há uma quantidade limitada de canais e poucos comandos não primordiais do controle remoto incluídos no sistema. Além disso, como citado nos resultados deste documento, entende-se que a forma de captura atual dos comandos enviados possui pouca confiabilidade, o que torna a elaboração de uma nova solução técnica para este fim, um interessante trabalho futuro. Há, ainda, a necessidade de integração com outros fabricantes de TV e conversores de sinal para tornar este sistema mais abrangente quanto aos usuários que ele satisfaz. Neste mesmo sentido, poderia-se buscar parcerias com os fabricantes de forma a conseguir maior integração técnica com seus dispositivos.

Na jornada do remédio, uma funcionalidade pensada pelo grupo mas não implementada foi a integração da jornada com o médico do idoso e farmácias locais, transmitindo as informações dos remédios tomados, podendo viabilizar a recomendação de doses, ou mesmo encomendar automaticamente a compra de novos medicamentos. Outra funcionalidade possível seria a de reconhecer o remédio inserido na caixa, como por código de barras da caixa ou observação das características do comprimido, transferindo a responsabilidade de informar o nome do usuário ao sistema. Uma melhoria possível para esta jornada, também, seria a respeito da detecção de que o remédio foi tomado, uma vez que isto, atualmente, é

feito baseado somente na abertura do compartimento, o que não necessariamente significa que o usuário de fato tomou o remédio.

Em relação ao *smart speaker*, possíveis trabalhos futuros seriam incluir serviços de *speech to text*, *NLU* e *text to speech* que funcionem localmente, sem a necessidade de conexão com a internet, adicionando maior disponibilidade e menor tempo de resposta ao dispositivo.

Quanto à inclusão de novas jornadas ao sistema, pensa-se na possibilidade de detectar quedas do usuário, pois este é um problema frequentemente enfrentado na terceira idade e responsável por muitos dos acidentes domésticos desta faixa etária. Para viabilizar e ajudar na comunicação do usuário com a família e amigos, poderia ser incluída uma jornada envolvendo vídeo conferências, tema que fez parte da discussão inicial deste trabalho. Quanto à alimentação, outro tópico importante para saúde e qualidade de vida, poderiam ser incluídas funcionalidades integradas à geladeira que viabilizassem a compra e reposição de alimentos.

Outro tema que fez parte do escopo inicial do projeto mas foi explorado menos que o planejado foi a capacidade do sistema de analisar e detectar anomalias na rotina do usuário que possam indicar situações críticas e de possível emergência. Algoritmos de inteligência artificial poderiam ser incluídos ao sistema, colhendo os dados das diferentes jornadas e analisando quebras estranhas de rotina que poderiam ser um indicativo de uma possível queda, problema de saúde, entre outras situações críticas. A partir desta detecção, ações reativas poderiam ser tomadas, tal como avisar familiares e amigos e acionar serviços de emergências. Além de esta inteligência ser uma ferramenta de prevenção e emergência, ela poderia ser explorada para recomendar produtos e atividades ao usuário, como sugerir programas de TV, alimentos, remédios e eventos.

REFERÊNCIAS

[1] PROJEÇÃO DA POPULAÇÃO DO BRASIL E DAS UNIDADES DA FEDERAÇÃO. IBGE. Disponível em: <https://www.ibge.gov.br/apps/populacao/projecao/>. Acesso em 15 abr. 2019.

[2] FOR the first time, 1 person in 5 in Japan is 70 or older. Japan Times, 2018. Disponível em: <https://www.japantimes.co.jp/news/2018/09/17/national/number-women-japan-aged-least-65-years-old-tops-20-million-first-time/>. Acesso em 15 abr. 2019.

[3] YASSUDA, Daniela; POSSANI, Hugo; MELO, Gabriela; HAYASHI, Victor; Hedwig - Casa Conectada. Escola Politécnica da USP. São Paulo. 2017

[4] CHEN, L.; NUGENT, C.D.; & WANG, H. (2012). A Knowledge-Driven Approach to Activity Recognition in Smart Homes. IEEE Transactions on Knowledge and Data Engineering, 24, pp. 961-974.

[5] PAL, DEBAJYOTI; FUNILKUL, SUREE; CHAROENKITKARN, NIPON; KANTHAMANON, PRASERT; Internet-of-Things and Smart Homes for Elderly Healthcare: An End User Perspective; Technology, King Mongkut's University of Technology Thonburi, Bangkok 10140, Thailand; IEEE SPECIAL SECTION ON HUMAN-CENTERED SMART SYSTEMS AND TECHNOLOGIES 2018;

[6] SPEECH RECOGNITION. GITHUB. Disponível em: <https://github.com/Uberi/speech_recognition>

[7] SNOWBOY, A CUSTOMIZABLE HOTWORD DETECTION ENGINE. KITT. Disponível em: <http://docs.kitt.ai/snowboy/#running-on-raspberry-pi>

[8] ANSARI, Junaid A., SATHYAMURTHY, Arasi,. BALASUBRAMANYAM, Ramesh. An Open Voice Command Interface Kit. 2015.

[9] YONEOKA, Naoki, ARAKAWA, Yutaka, YATSUMOTO, Keiichi. Detecting Surrounding Users by Reverberation Analysis with a Smart Speaker and Microphone Array. 2019.

[10] PROJECT ALIAS. VIMEO. Disponível em: <https://vimeo.com/306044007>

[11] HAYASHI, Victor. Manual Kit Acelerador IoT Hedwig.

[12] AL-FUQAHA, A.; GUIZANI, M.; MOHAMMADI, M.; ALEDHARI, M.; & AYYASH, M. Internet of Things: A Survey on Enabling Technologies, Protocols, and

Applications. IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347-2376, 2015.

[13] XU, L. D.; HE, W.; & LI, S. Internet of Things in Industries: A Survey. IEEE Transactions on Industrial Informatics, vol. 10, no. 4, pp. 2233-2243, Nov. 2014.

[14] RASPBERRY PI 3 SPECIFICATIONS. Disponível em: <https://magpi.raspberrypi.org/articles/raspberry-pi-3-specs-benchmarks>

[15] GUIDE TO THE TIVA TM4C123. LAUNCH PAD. Disponível em: <https://energia.nu/pinmaps/ek-tm4c123gxl/>

[16] CLOUD SPEECH-TO-TEXT. GOOGLE. Disponível em: <https://cloud.google.com/speech-to-text/>

[17] WATSON SPEECH TO TEXT. IBM. Disponível em: <https://www.ibm.com/watson/services/speech-to-text/>

[18] DIALOGFLOW. Disponível em: <https://dialogflow.com/>

[19] RASA: OPEN SOURCE CONVERSATIONAL AI - RASA. RASA. Disponível em: <https://rasa.com/>

[20] CLOUD TEXT-TO-SPEECH. GOOGLE. Disponível em: <https://cloud.google.com/text-to-speech/>

[21] AMAZON POLLY. AMAZON. Disponível em: <https://aws.amazon.com/polly/>

[22] SOA FUNDAMENTALS IN A NUTSHELL. IBM. Disponível em: <https://www.ibm.com/developerworks/webservices/tutorials/ws-soa-ibmcertified/ws-soa-ibmcertified.html>

[23] MONJA COEN: CONFIANÇA (VISÃO ZEN BUDISTA). YOUTUBE. Disponível em: <https://www.youtube.com/watch?v=DUGGICnIYws&t=148s>

[24] CHICO BUARQUE LÊ TRECHO DE "BUDAPESTE". YOUTUBE. Disponível em: <https://www.youtube.com/watch?v=WXLFXX0WA1Y&t=39s>

[25] DISCURSO DE CHARLES CHAPLIN EM "O GRANDE DITADOR". YOUTUBE. Disponível em: <https://www.youtube.com/watch?v=geOQWt5tsbY>

[26] ROGERIO CENI - DISCURSO. YOUTUBE. Disponível em: <https://www.youtube.com/watch?v=j3NsL8rWQI4>

[27] SPEEDTEST. Disponível em: <https://www.speedtest.net/result/8488261436>

APÊNDICE A - ENTREVISTA COM POSSÍVEL USUÁRIO

Entrevistador: Primeiro gostaria que vocês falassem os seus nomes e suas idades.
Kiyoko: Meu nome é Ilda Kiyoko, tenho 61 anos.
Fumiyo: Meu nome é Yamada Fumiyo, tenho 94 anos.

Entrevistador: Estamos abordando duas rotinas de idosos, fazendo automatização destas para facilitar o dia dia. Uma delas é o hábito de assistir TV e a outra é a rotina de medicamentos.

Entrevistador: Em relação a TV, qual a maior dificuldade que a dona Fumiyo sente no dia dia?
Kiyoko: Mudar de canal para o que ela assiste.
Entrevistador: Ela segue uma rotina bem definida em relação ao que ela assiste?
Kiyoko: Ela sempre assiste assiste o programa de música japonesa. Toda semana temos que ligar no canal para ela.

Entrevistador: E em relação aos medicamentos que dificuldades vocês sentem?
Kiyoko: Ela não lembra mais de tomar remédio. E sempre tenho lembrar e dar o remédio para ela. E as vezes eu também esqueço e acaba atrasando, então tem dia que ela toma as 8 horas, tem dia que ela toma as 9. Mas como eu anoto em uma agenda todo dia que hora ela tomou conseguimos controlar isso.

Entrevistador: Então a automatização dessas tarefas ajudariam muito nas suas rotinas?
Kiyoko: Nossa, ajudaria muito. Até se ela pudesse só falar para TV desligar, pra ela seria muito mais fácil.

Entrevistador: Muito obrigado pelo tempo de vocês.
Kiyoko: De nada.
Fumiyo: De nada

APÊNDICE B - REPOSITÓRIOS

Aplicativo:
https://github.com/veraldo/smart-home-app

Smart speaker:
https://github.com/ViniciusGarciaSilva/speaker-API

TV - API:
https://github.com/ViniciusGarciaSilva/tv-routine-api

TV - Banco de dados:
https://github.com/ViniciusGarciaSilva/tv-routine-db

Remédios - Banco de dados:
https://github.com/veraldo/medicine-api

Hub
https://github.com/ViniciusGarciaSilva/hub