UNIFACS UNIVERSIDADE SALVADOR
PROGRAMA DE PÓS GRADUAÇÃO EM SISTEMAS E COMPUTAÇÃO
MESTRADO EM SISTEMAS E COMPUTAÇÃO

ÍTALO RIBEIRO COSTA DOS SANTOS

UM ESTUDO SOBRE A EFICÁCIA DE UMA ABORDAGEM HÍBRIDA PARA RECONHECIMENTO DE VOZ

Salvador
2016

ÍTALO RIBEIRO COSTA DOS SANTOS

UM ESTUDO SOBRE A EFICÁCIA DE UMA ABORDAGEM HÍBRIDA PARA RECONHECIMENTO DE VOZ

Dissertação apresentada ao Mestrado Acadêmico em Sistemas e Computação da UNIFACS Universidade Salvador, Laureate International Universities, como requisito parcial para obtenção do título de Mestre.

Orientador: Prof. Dr. Artur Henrique Kronbauer.

Salvador
2016

FICHA CATALOGRÁFICA
Elaborada pelo Sistema de Bibliotecas da UNIFACS Universidade Salvador, Laureate Internacional Universities

Santos, Ítalo Ribeiro Costa dos

Um estudo sobre a eficácia de uma abordagem híbrida para reconhecimento de voz. / Ítalo Ribeiro Costa dos Santos.- Salvador, 2016.

174 f.: il.

Dissertação apresentada ao Curso de Mestrado em Sistemas e Computação, UNIFACS Universidade Salvador, Laureate Internacional Universities, como requisito parcial para obtenção do grau de Mestre.

Orientador Prof. Dr. Artur Henrique Kronbauer.

1. Sistema de processamento da fala. 2. Voice Home. I. Kronbauer, Artur Henrique, orient. II. Título.

CDD: 006.454

TERMO DE APROVAÇÃO

ÍTALO RIBEIRO COSTA DOS SANTOS

UM ESTUDO SOBRE A EFICÁCIA DE UMA ABORDAGEM HÍBRIDA PARA RECONHECIMENTO DE VOZ

Dissertação aprovada como requisito parcial para obtenção do título de Mestre em Sistema de Computação, UNIFACS Universidade Salvador, Laureate International Universities, pela seguinte banca examinadora:

Artur Henrique Kronbauer – Orientador _____
Doutor em Ciência da Computação pela Universidade Federal da Bahia (UFBA)
UNIFACS Universidade Salvador, Laureate International Universities

Joberto S. B. Martins _____
Doutor em Computação (PhD) pela Université Pierre et Marie Curie, França
UNIFACS Universidade Salvador, Laureate International Universities

Daniel Barbosa _____
Doutor em Engenharia Elétrica pela Universidade de São Paulo - USP
Universidade Federal da Bahia - UFBA

Salvador, 11 de novembro de 2016.

Dedico esta dissertação a minha esposa Mizure e aos meus irmãos Átila e Erlon e aos pais Eron e Heliane, pelo apoio e estrutura oferecida.

AGRADECIMENTOS

Agradeço a meus pais, Eron e Heliane, pelo suporte e apoio incondicional, além da minha família e amigos pelo apoio e compreensão nesta fase.

Agradeço a meus irmãos, Átila e Erlon, que apesar das desavenças sempre estão presentes na minha vida.

Um agradecimento especial ao meu orientador, pois é uma pessoa muito amiga e sempre estava a minha disposição, mesmo nos momentos mais difíceis. Além disso, sempre foi uma pessoa compreensiva e paciente.

Gostaria de registrar meus agradecimentos à equipe de pesquisa do Framework Para Ambientes Inteligentes: Igor Pimenta, Fabio Gomes, Brunno Brito de Araújo e Lívia Sarmento.

“O poder nasce do querer. Sempre que o homem aplicar a veemência e perseverante energia de sua alma a um fim, vencerá os obstáculos, e, se não atingir o alvo fará, pelo menos, coisas admiráveis”.

José de Alencar.

RESUMO

Na última década, observou-se um progresso significativo nas tecnologias de reconhecimento de voz com aplicação em Ambientes Inteligentes (AmI). A possibilidade de utilizar a voz como forma de interação possibilita integrar as pessoas ao cenário que as cerca de forma natural e transparente. Entretanto, ainda existem obstáculos a serem transpostos, tais como, o tratamento de expressões genéricas, as variações de acordo com o perfil dos usuários, timbre de voz e comportamento espontâneo. Para contribuir com a solução de alguns destes problemas, é proposto neste trabalho uma modelo para permitir interações naturais via voz, abrangendo uma abordagem híbrida e adaptativa ao perfil dos usuários. Para validar o modelo, foi criada uma plataforma, chamada de Voice Home, capaz de interpretar comandos frequentes, em uma estrutura local, e novos comandos, em uma estrutura em nuvem. De acordo com os testes de usabilidade, foi observado que existem ganhos significativos com a abordagem híbrida, possibilitando adaptação do sistema ao perfil dos usuários.

Palavras chave: Ambientes Inteligentes. Interações Naturais. Interação Via Voz. Casas Inteligentes. Modelo. Plataforma. Experimento. Voice Home.

ABSTRACT

In the last decade, there has been significant progress in voice recognition technologies with applications in Intelligent Environments (AmI). The possibility of using the voice as a form of interaction enables people to integrate the scenario that some natural and transparent manner. Entreated, there are still obstacles to be overcome, such as the treatment of general expressions, variations according to the profile of users, voice timbre and spontaneous behavior. To contribute to solving some of these problems, this work proposes a model to allow natural interactions via voice, including a hybrid and adaptive approach to profile users. To validate the model, a platform capable of interpreting frequently used commands was created in a local structure, and new commands in a cloud structure. According to usability testing, it was observed that there are significant gains to the hybrid approach, allowing adaptation of the system to the user's profile.

Key Words: Ambient Intelligence. Natural Interaction. Voice Based Interaction. Smart Home. Model. Platform. Experiment.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACM	Association for Computer
AmI	Ambient Intelligent
ANN	Artificial Neural Networks
ANOVA	Analysis of Variance
AR	Automação Residencial
ASR	Automatic Speech Recognition
AT&T	American Telephone and Telegraph Corporation
DECIDE	Determine Explore Choose Identify Decide Evaluate
DHCP	Protocolo de configuração dinâmica de host
DSR	Design Science Reseach
HMM	Hidden Markov Model
HTK	Hidden Markov Model Toolkit
IBGE	Instituto Brasileiro de Geografia e Estatística
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IHC	Interação Humano-Computador
IP	Internet Protocol
IPEA	Instituto de Pesquisa Econômica Aplicada
JSON	JavaScript Object Notation
MLP	Multi Layer Perceptron
NUI	Natural User Interface
RNA	Redes Neurais Artificiais
RV	Reconhecimento de Voz
SOA	System Oriented Architechture
TPF	Texto para Fala
TTS	Text-to-Speech
UML	Unified Modeling Language
Wi-Fi	Wireless Fidelity

SUMÁRIO

1 INTRODUÇÃO

O Reconhecimento Automático da Fala (*Automatic Speech Recognition* – ASR) é um assunto que tem sido alvo de grandes discussões em diversos trabalhos científicos. Isto se deve ao fato de que a voz é uma forma natural de interação e envolve uma grande variedade de opções para integrá-la em um cenário de interação (HAMILL et al., 2009).

Grande parte das abordagens científicas reconhecem as dificuldades enfrentadas por pessoas com necessidades especiais ou mesmo idosos e relacionam o uso da voz como uma forma de autonomia (LECOUTEUX et al., 2011). Algumas iniciativas discutem tais dificuldades e os principais fatores envolvidos, outras buscam, por meio de metodologias, propor formas criativas de mitigar tais problemas. Existem também aquelas que buscam, modelos para automação para setores como, por exemplo, telecomunicação, saúde, educação e automação residencial (MARELI et al., 2013).

Independente da abordagem utilizada, todos os estudos advogam pela melhoria do processo de reconhecimento automático da fala e reconhecem que ainda existem muitas oportunidades para contribuições nesta área. Neste sentido, está dissertação objetiva, por meio da pesquisa bibliográfica na área de reconhecimento automático da fala, formar a base de conhecimentos necessários para a definição de um estudo sobre uma abordagem eficiente para o reconhecimento de voz em ambientes inteligentes. A proposta é alinhar metodologias distintas para a formação de um modelo híbrido, contribuindo para melhorar a eficiência do reconhecimento dos comandos proferidos pelos usuários neste cenário.

1.1. CONTEXTO

A proposta apresentada nesta dissertação está inserida em uma abordagem híbrida para reconhecimento automático da fala. A qual pode ser aplicada em duas áreas: (I) **reconhecimento local** e (II) **reconhecimento em nuvem**. A estratégia é aliar o tempo de resposta das ferramentas locais com o melhor desempenho de

interpretação das ferramentas disponíveis em nuvem, tornando o vocábulo adaptável ao perfil do usuário.

1.2. PROBLEMA

O estudo da bibliografia científica, realizado neste trabalho, sobre o reconhecimento automático da fala como meio de interação entre seres humanos e dispositivos computacionais apresentaram evidências de que:

- A variação fonética impõe o crescimento do processamento com o aumento do vocábulo para reconhecimento. López and Callejas (2010) afirmam que a grande heterogeneidade de palavras envolvidas na tarefa do reconhecimento de voz a torna complexa.

- Existência de problemas ocultos de ruído na acústica do ambiente (Lecouteux et al., 2001). Segundo Vacher et al. (2014) é preciso considerar o ruído, por menor que seja, em qualquer abordagem envolvendo interações com a voz.

- A qualidade do reconhecimento esta correlacionada ao desempenho do sistema. Os níveis de dificuldade e consequentemente, a necessidade de mais processamento é proporcional ao tamanho do vocábulo utilizado (PORTET et al., 2014).

- Há necessidade de abordagens mais precisas para o reconhecimento automático da fala que atendam a utilização da linguagem natural. Cerón e Badillo (2011) apresentam dados estatísticos que comprovam as dificuldades para que o vocabulário de palavras decodificadas pelos computadores sejam estendidos, permitindo que as palavras proferidas sejam interpretadas como comandos em um espetro amplo, atendendo dessa forma, o vasto vocabulário utilizado pelas pessoas.

De forma resumida, entende-se que a diversidade linguística obriga a utilização de vocábulos grandes e genéricos para reconhecimento. Este desafio, aliado ao ruído e limitações do reconhecimento local, prejudica o desempenho do sistema com o

aumento exacerbado do processamento, causando uma depreciação na qualidade do reconhecimento, principalmente, quanto a assertividade e tempo de resposta.

Com relação às possíveis soluções para os problemas apresentados, Lecouteux et al. (2001) propõem várias adaptações no cenário de interação para diminuir a degradação do sistema de reconhecimento de voz, tais como:

(i) Diminuir à distância de quem fala e o microfone.

(ii) Sobreposição de sinais vindos de múltiplos microfones.

(iii) Utilizar mais de um tipo de ferramenta de tradução de voz, de maneira concorrente, para diminuir os erros.

Com relação ao cenário de interação, contextualizado em ambientes inteligentes e pautados nas premissas da Computação Ubíqua e Pervasiva, Mareli et al. (2013) apontam que várias restrições podem interferir no funcionamento adequado dos dispositivos envolvidos:

(i) Dificuldade em se estabelecer um protocolo comum de comunicação, por conta da heterogeneidade dos dispositivos encontrados no cenário.

(ii) Complexidade dependente da quantidade e variedade de informações de contexto a serem tratadas.

(iii) Dificuldade para o desenvolvimento e teste de aplicações, já que exige cenários instrumentados com dispositivos computacionais específicos, tais como, sensores e atuadores.

(iv) Necessidade de alto desempenho da rede de dados para permitir a troca de informações em tempo real.

A partir da identificação da problemática em questão e da percepção da importância de um reconhecimento eficaz para os processos da civilização moderna, foi possível formular o problema a ser investigado nessa dissertação: **Como tornar mais**

eficiente, adaptável e menos oneroso o reconhecimento automático da fala, devido às dificuldades enfrentadas na qualidade de interação e comunicação?

1.3. MOTIVAÇÕES

As motivações para a escolha do reconhecimento automático da fala como área de pesquisa, está pautado em estudos como, por exemplo, de Hamill et al. (2009) que destacam o reconhecimento de comandos de voz como sendo a mais promissora tecnologia empregada para a interação natural entre seres humanos e dispositivos computacionais.

Segundo Liu (2010) a Interface Natural de Usuário (*Natural User Interface* – NUI), baseada em voz, torna a experiência no processo de comunicação entre seres humanos e dispositivos computacionais mais natural e confiável, sendo reconhecida como um canal conveniente e eficiente para a troca de informações.

Portet et al. (2013) afirmam que as interfaces via voz são mais adaptativas do que interfaces táteis, que precisa de interação física e visual, além de facilitar a interação de pessoas idosas e com restrições de movimento.

Outros estudos mais antigos, como os de Gárate et al. (2005), Yi et al. (2007) e Gao et al. (2007) apresentaram a associação do ganho de autonomia, por parte dos idosos, com os processos relacionados a interações por voz.

1.4. JUSTIFICATIVA

As dificuldades identificadas no estudo do reconhecimento automático da fala, aliado a necessidade de aprimorar a interação via comandos de voz em ambientes ubíquos e pervasivos são de extrema importância no cenário atual da computação (BOUAKAZ et al., 2014).

Na revisão bibliográfica sobre o reconhecimento automático da fala, foi possível constatar a importância do estudo da voz como meio natural de interação, sendo

assim, uma importante modalidade de interação entre o ser humano e dispositivos computacionais (VACHER et al., 2015).

Neste contexto, destacam-se alguns trabalhos, tais como, os de Liu (2011), Blake (2011) e Wigdor and Wixon (2011) que descreveram a voz como um canal natural e confiável para comunicação entre seres humanos e dispositivos computacionais. Lecouteux et al. (2011) forneceram evidências científicas acerca de métodos artificiais para reconhecimento da voz, e Portet et al. (2013), descreveram a voz como um mecanismo de processamento de informações para a utilização em uma vasta gama de aplicações. Outros trabalhos importantes investigados foram os de Vacher et al. (2016) e Chahuara et al. (2016), que relatam avanços científicos nos experimentos com o uso da voz cujos resultados ratificam as evidências científicas sobre as contribuições apresentadas por Lecouteux.

Outo ponto importante discutido na literatura, são as vantagens e desvantagens de se utilizar uma plataforma local ou em nuvem para a tradução dos comandos. Segundo Morbini et al. (2013) o reconhecimento local é mais rápido e apropriado para a execução de comandos em tempo real. Por outro lado, os autores afirmam que uma plataforma em nuvem pode disponibilizar mais recursos computacionais, o que permite trabalhar com vocábulos mais abrangentes, sendo ideal para a tradução dos comandos proferidos por seres humanos.

Neste contexto, realizar uma pesquisa que possa desfrutar das vantagens de ambas as abordagens é uma excelente justificativa para a realização de um estudo nesta área. Inclusive, com a proposta de uma nova abordagem híbrida, que possa melhorar a experiência dos usuários, garantir altas taxas de reconhecimento e personalizar a base de conhecimento de acordo com o perfil dos usuários de um determinado cenário de interação.

1.5. OBJETIVO

O objetivo geral deste trabalho é propor um modelo híbrido e adaptativo de reconhecimento automático da fala, para desenvolvimento de aplicações com interfaces naturais. Para alcançar este objetivo foram definidos os seguintes objetivos específicos:

- Realizar um levantamento bibliográfico para identificar as particularidades de trabalhos que relatam o uso de reconhecimento automático da fala como meio de interação entre seres humanos e dispositivos computacionais.
- Usar as constatações obtidas com o estudo do estado da arte como fundamentação científica para a definição do modelo híbrido.
- Implementar uma plataforma para validar o modelo proposto.
- Realizar um estudo de caso com o intuito de avaliar a plataforma e identificar os pontos positivos e negativos da proposta.

1.6. CONTRIBUIÇÕES

Em conformidade com os objetivos gerais e específicos previstos, este trabalho propõe um modelo que é validado com a construção de uma plataforma para o reconhecimento e execução de comandos de voz em ambientes ubíquos e pervasivos. A plataforma desenvolvida disponibiliza um *middleware* com as seguintes características:

- Reconhecer de forma dinâmica diferentes tipos de dispositivos eletrônicos, independente do modelo ou marca.

- Empregar uma forma híbrida para interpretar os comandos de voz, utilizando-se de um sistema local para a tradução de comandos já conhecidos e um sistema em nuvem quando o reconhecimento local não for satisfatório.

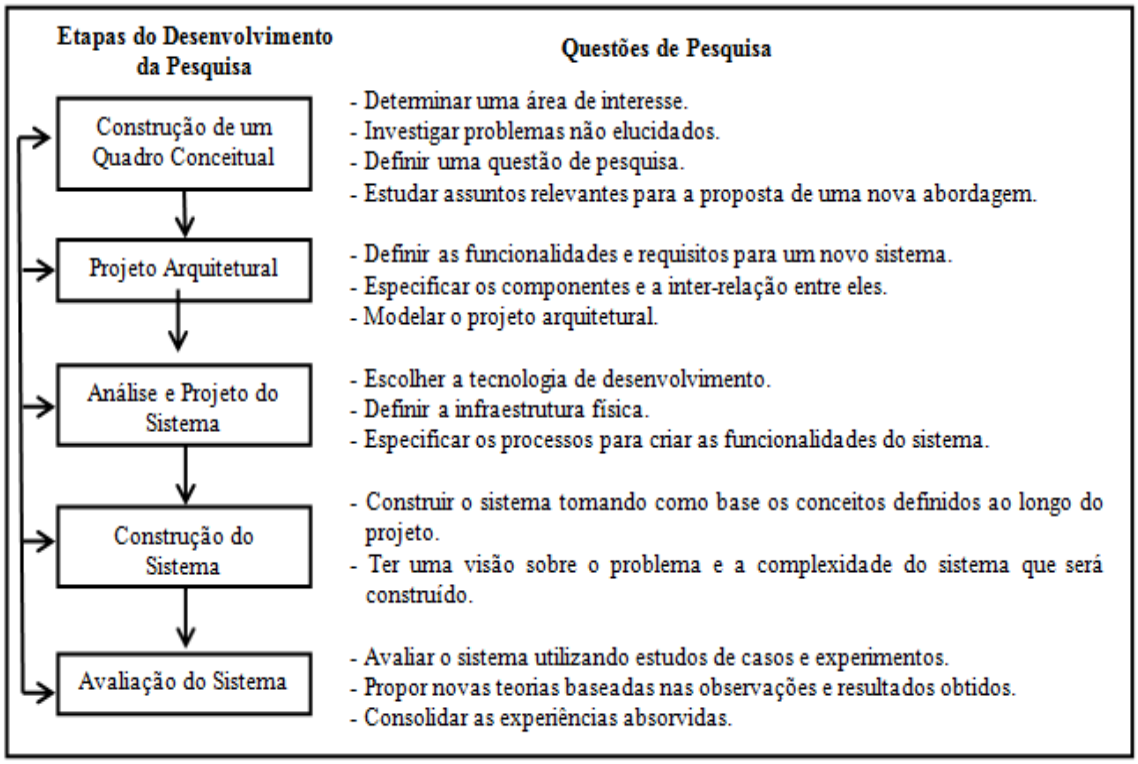
- O vocabulário de reconhecimento será adaptável ao perfil do usuário, já que a base local será atualizada todas as vezes que a tradução em nuvem for acionada.

- O modelo foi proposto em camadas, desta forma, pode ser implementado com diferentes tipos de hardwares ou softwares.

1.7. METODOLOGIA

A metodologia *Design Science Reseach* (DSR), proposta por Hevner e Chatterjee (2010), foi utilizada para direcionar as etapas executadas durante a realização dessa tese. Essa metodologia contempla cinco fases distintas, conforme pode ser observado na Figura 1.

Figura 1 - Etapas do desenvolvimento do projeto de pesquisa



Fonte: Hevner e Chatterjee (2010).

A Tabela 1 apresenta uma visão geral de como o andamento deste trabalho se integra com as etapas do desenvolvimento do projeto descrito por Hevner e Chatterjee (2010). Nas próximas seções deste capítulo, será aprofundada a correlação da metodologia de pesquisa DSR com as etapas desenvolvidas ao longo do presente estudo.

Tabela 1 - Relação das etapas da metodologia DSR com a presente pesquisa

Etapas de Desenvolvimento	Questões de Pesquisa
Construção de um Quadro Conceitual	✓ Estudo aprofundado da literatura com a intenção de identificar pontos ainda não explorados na área de interação via voz em Ambientes Ubíquos e Pervasívos. ✓ Investigar as funcionalidades das abordagens existentes com a intenção de definir uma nova abordagem para o reconhecimento de voz em meio híbrido, atuando em nuvem e local, complementando propostas atuais. ✓ Estudo de disciplinas relevantes que possam ser utilizadas para subsidiar a criação de uma nova abordagem.
Projeto Arquitetural	✓ Definir o modelo de uma nova abordagem para avaliar o desempenho de comandos de voz em Ambientes Ubíquos e Pervasivos, abrangendo sua arquitetura, modularização e extensibilidade. ✓ Demarcar as funcionalidades de cada componente do modelo e as inter-relações entre eles.
Análise e Projeto do Sistema	✓ Projetar uma plataforma que possa contemplar as especificações do modelo. ✓ Definir os processos envolvidos na criação da plataforma. ✓ Especificar soluções tecnológicas e escolher as mais convenientes para cada componente representado na plataforma.
Construção do Sistema	✓ Construir uma plataforma que tem como base os conceitos definidos nas fases anteriores. ✓ Elaborar uma prova de conceito para validar a plataforma em um Ambiente Ubíquo e Pervasívo baseado em interações via voz.
Avaliação do Sistema	✓ Especificar e executar um experimento que possa validar as funcionalidades da plataforma e descrição dos processos envolvidos para a sua utilização. ✓ Consolidar as experiências adquiridas por intermédio da descrição dos resultados obtidos no experimento.

Fonte: Elaborado pelo autor (2016).

1.7.1. Construção de um Quadro Conceitual

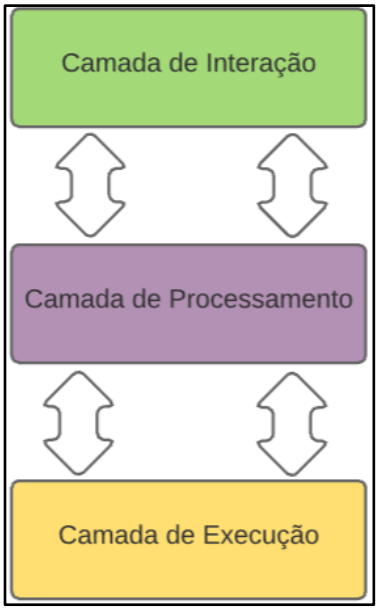
O conhecimento do estado da arte, no campo específico da investigação, é obtido a partir da leitura de pesquisas semelhantes, publicadas em anais de conferências e periódicos, sendo essencial para a contextualização do problema e o posicionamento do trabalho diante da comunidade, como será apresentado no Capítulo 3 desta dissertação.

Uma visão aprofundada sobre o tema de estudo é importante para reforçar o entendimento de quais problemas foram sanados anteriormente e dos que ainda estão em aberto. É importante que os pesquisadores estejam seguros de que a pesquisa contém novos aspectos ainda não abordados, evitando perda de energia com problemas já debatidos anteriormente e solucionados (HANSEN, 2012).

1.7.2. Projeto Arquitetural

A abordagem utilizada é denominada de modelo três camadas, que é derivada do modelo multicamadas (EDWARDS; FOREWORD-BY-ORFALI, 1998). Cada camada trata de um conjunto de objetivos específicos do projeto arquitetural e funciona de forma independente, na qual possa ser substituído sem prejuízo para a plataforma. Para contemplar a descrição do modelo proposto nesta dissertação a
Figura **2** é uma adaptação da proposta original de Edward e Foreword-By-Orfali (1998).

Figura 2 - Modelo Proposto



Fonte: Adaptado de Edwards e Foreword By-Orfali (1998).

Os pesquisadores descrevem suas camadas em três níveis. As camadas propostas pelo modelo são:

- Camada de Interação: possibilita a execução de comandos provenientes dos usuários, sendo responsável por prover mecanismos de captura das interações.
- Camada de Processamento: responsável por traduzis as requisições dos usuários em ações que devem ser executadas no cenário de interação. Pode

também ser chamada de camada intermediária, servidor de aplicação ou *middleware*.

- Camada de Execução: responsável por receber as requisições da camada de processamento, para que possa executar as funcionalidades do sistema.

1.7.3. Análise e Projeto do Sistema

Para a realização deste levantamento, foram especificados dois pontos de interesse: (i) as linguagens em que os componentes são desenvolvidos; e (ii) as ferramentas para dar suporte a interpretação dos comandos de voz.

Os componentes das camadas de Interação e Processamento foram desenvolvidos em Python com suporte à BBC BASIC. A Linguagem é simples, objetiva, disponibilizada gratuitamente e fortemente acoplável a dispositivos embarcados. A *Integrated Development Environments* (IDE) utilizada no projeto foi a IDLE, um ambiente de desenvolvimento integrado para Python.

A linguagem Java, na versão 7.0, foi escolhida para o desenvolvimento do módulo de interligação dos atuadores. A sua adoção foi em função de ser disponibilizada gratuitamente e compatível com dispositivos embarcados. A IDE utilizada para apoiar o desenvolvimento em Java foi o Eclipse na versão Kepler Service Release 1.

Para a implementação dos componentes da camada de execução foi utilizada a linguagem C++, uma vez que essa linguagem disponibiliza bibliotecas específicas para a manipulação de microcontroladores. A IDE utilizada para o desenvolvimento foi a open-source Arduíno Software, disponibilizada gratuitamente.

O reconhecimento de voz é feito através de um microfone de eletreto, que é conectado ao Raspberry PI via USB para a aquisição do sinal de voz do usuário, sendo destinado a converter as vibrações sonoras em sinais elétricos para análises dos componentes locais e em nuvem.

Para o componente local foi utilizado o algoritmo Julius, que utiliza a tecnologia baseada em áudio. Além disso, essa escolha se deu por uma série de razões, tais

como: (i) possui uma plataforma estruturada para esta finalidade possibilitando o reconhecimento da linguagem portuguesa; (ii) acoplado a um microfone, não depende de nenhum tipo de acessório para que a voz seja capturada; e (iii) é disponibilizada gratuitamente;

O componente em nuvem utiliza a API Google Speech. A escolha por esta plataforma foi em função dos seguintes pontos: (i) é umas das tecnologias pioneira em nuvem para tratamento de áudio; (ii) interpreta a língua portuguesa; (iii) apresenta bom desempenho; e (iv) sua utilização é fácil.

O processamento será em nuvem apenas quando o reconhecimento local não for satisfatório. No Capítulo 4, será descrito o modelo que foi idealizado para a captura e reconhecimento de voz.

1.7.4. Construção do Sistema

Nesta fase, os trabalhos direcionaram-se para a elaboração de processos que auxiliaram a codificação da abordagem proposta na pesquisa. A Engenharia de Software utiliza um número razoável de técnicas relacionadas com o desenvolvimento controlado de sistemas. Com base nessas técnicas, os desenvolvedores devem ser capazes de lidar com a vasta gama de informações que compreendem o desenvolvimento de uma plataforma. Além disso, é de suma importância que esses desenvolvedores sejam capazes de organizar o trabalho e os recursos envolvidos no processo de codificação (PHAM ; PHAM, 2011).

O processo de desenvolvimento de uma plataforma é definido como o conjunto de elementos responsáveis por transformar os requisitos dos usuários em uma solução computacional. Assim, o processo é basicamente composto por um conjunto ordenado de tarefas que estão relacionadas com uma série de outros elementos. Por exemplo, os recursos consumidos e modificados por essas tarefas, os produtos obtidos com a realização dessas tarefas, os procedimentos e condutas seguidos durante a execução, entre outros (HIRAMA, 2012).

Para descrição e controle dos passos para desenvolvimento da codificação dos componentes abordados nesta dissertação, foi escolhido o Lucidchart[1], o qual possui integração com as principais linguagens de programação, contemplando todos os aspectos necessários para o desenvolvimento do projeto.

1.7.5. Avaliação do Sistema

O experimento realizado como parte dos estudos apresentados nesta pesquisa utiliza uma abordagem quantitativa abrangendo a avaliação da usabilidade e desempenho da plataforma proposta. Neste contexto, foi realizado um estudo de caso com estudantes da Universidade Salvador, para avaliar na prática a eficiência da abordagem proposta nesta pesquisa.

O experimento abrangeu duas técnicas de avaliação (questionário e observação direta) para avaliar os atributos de usabilidade definidos por Kronbauer e Santos (2013) com o intuito de estimar as potencialidades e limitações das interações na plataforma proposta, com usuários reais, nos mais variados objetos eletrônicos.

Os passos realizados durante todas as fases do experimento foram demarcados pelo *framework* DECIDE proposto por Preece et al. (2015). O DECIDE orienta o planejamento, a execução e a análise de uma avaliação de IHC. As atividades do *framework* são interligadas e executadas iterativamente à medida que o avaliador articula os objetivos da avaliação, os dados e recursos disponíveis. As atividades do *framework* estão descritas na Tabela 2 e no capítulo 5, serão apresentados os resultados e análises obtidas com o estudo de caso, tomando como direcionamento os passos do *framework*.

[1]Disponível em: https://www.lucidchart.com/pt

Tabela 2 - Atividades do framework DECIDE

D	Determinar os objetivos da avaliação e identificar por que e para quem tais objetivos são importantes.
E	Explorar perguntas a serem respondidas com a avaliação. Para cada objetivo definido, o avaliador deve elaborar perguntas específicas a serem respondidas durante a avaliação.
C	Escolher (*Choose*) os métodos de avaliação a serem utilizados. O avaliador deve escolher os métodos mais adequados para responder às perguntas e atingir os objetivos esperados, considerando também o prazo, o orçamento, os equipamentos disponíveis e o grau de conhecimento e experiência dos avaliadores.
I	Identificar e administrar as questões práticas da avaliação como, por exemplo, o recrutamento dos usuários que participarão da avaliação, a preparação e o uso dos equipamentos necessários, os prazos e o orçamento disponível, além da mão de obra necessária para conduzir a avaliação.
D	Decidir como lidar com as questões éticas. Sempre que usuários são envolvidos numa avaliação, o avaliador deve tomar os cuidados éticos necessários.
E	Avaliar (*Evaluate*), interpretar e apresentar os dados. O avaliador deve considerar: o grau de confiabilidade dos dados; se o método de avaliação mede o que deve medir; se os resultados podem ser generalizados; e o quanto os materiais, métodos e ambiente de estudo se assemelham à situação real investigada.

Fonte: Preece *et al.* (2015).

1.8. ESTRUTURA DA DISSERTAÇÃO

Com o intuito de descrever as etapas que compreendem o desenvolvimento dessa pesquisa, definimos a estrutura do trabalho da seguinte forma:

- Capítulo 1 – Contextualiza e descreve os motivos para a realização deste trabalho.
- Capítulo 2 – Apresenta as definições de computação ubíqua e a sua importância na busca do entendimento de uma interface inteligente.
- Capítulo 3 – Discute aspectos dos processos de reconhecimento automático da fala.

- Capítulo 4 – Apresenta um levantamento bibliográfico sobre o uso do reconhecimento de voz aplicada a ambientes inteligentes.
- Capítulo 5 – Especifica o modelo híbrido proposto para o reconhecimento automático da fala, com base na sumarização do levantamento bibliográfico descrito no capítulo 3 e apresenta os detalhes da implementação deste modelo.
- Capítulo 6 – Descreve os passos para a execução de um experimento com o objetivo de avaliar a plataforma proposta e verificar as suas potencialidades para ser adotada em larga escala. Além disso, discorre sobre os resultados do estudo de caso.
- Capítulo 7 – Apresenta as conclusões e relata as possibilidades de trabalhos futuros.

2 INTERFACES ADAPTÁVEIS

Neste capítulo, são abordados os principais tópicos e conceitos relacionados ao tema principal deste trabalho. Desta forma, são apresentados alguns conceitos clássicos de computação ubíqua, internet das coisas, ambientes inteligentes e interações naturais via voz.

2.1. COMPUTAÇÃO UBÍQUA

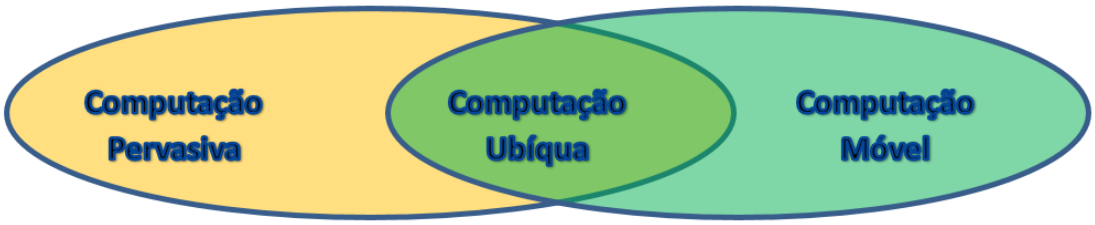
A Computação Ubíqua está alicerçada na ideia de que os computadores estarão em todos os lugares e em todos os momentos auxiliando o ser humano sem que ele tenha consciência disso. Desta forma, é um paradigma no qual a computação é profundamente integrada, de modo transparente às atividades cotidianas dos usuários (WEISER, 1991).

Uma das principais características dos sistemas ubíquos são os ambientes altamente dinâmicos nos quais tais sistemas são inseridos. Nestes ambientes, vários dispositivos interagem entre si para fornecer informações relevantes que contribuam com as atividades diárias de seus usuários de modo imperceptível (COOK et al., 2009).

Um dos propósitos da Computação Ubíqua é facilitar a interação entre usuários e dispositivos computacionais, de modo que cada usuário não perceba que está dando comandos aos computadores espalhados pelo ambiente que os cerca. Para proporcionar esta funcionalidade, os sistemas ubíquos capturam informações sobre o ambiente para dinamicamente se adaptar ao contexto e automaticamente executar ações apropriadas a cada mudança no cenário de interação (COOK et al., 2009).

Segundo previsões de Waiser (1991) estamos em uma nova era na área da computação, onde uma das principais características é que as interações com dispositivos computacionais ocorrerão de forma natural, da mesma forma como nos comunicamos com outros seres humanos. Uma aplicação ubíqua idêntica as necessidades de seus usuários coletando, por meio de sensores, as informações do seu contexto de execução, e as atende provendo serviços, por meio de atuadores, os quais incluem diversos tipos de interfaces (COOK et al., 2009) (Figura 3).

Figura 3 - Áreas que formam a Computação Ubíqua



Fonte: Cook et al. (2009).

A Computação Pervasiva prevê que os componentes computacionais estarão embarcados no ambiente de forma invisível para o usuário, tendo a capacidade de obter informações acerca do cenário e utilizá-la para controlar, configurar e ajustar as aplicações para melhor se adequar ao ambiente.

O ambiente também deve ser capaz de detectar outros dispositivos que adentrem a ele. Dessa interação surge a capacidade de computadores agirem de forma “inteligente” no cenário em que o usuário se locomove.

A Computação Móvel permite que seus usuários tenham acesso a serviços independentemente de sua localização podendo, inclusive, estar em movimento. Tecnicamente, é um conceito que envolve processamento, mobilidade e comunicação sem fio. A ideia é ter acesso à informação em qualquer lugar e a qualquer momento. Além da mobilidade física dos dispositivos, os softwares deverão ser dotados de mobilidade lógica.

2.1.1. Propriedades da Computação Ubíqua

A Computação Ubíqua está fundamentada em nove propriedades: integração física, invisibilidade, pró-atividade, sensibilidade ao contexto, interoperabilidade espontânea, interfaces naturais, coordenação, adaptação e tolerância a falhas e mobilidade.

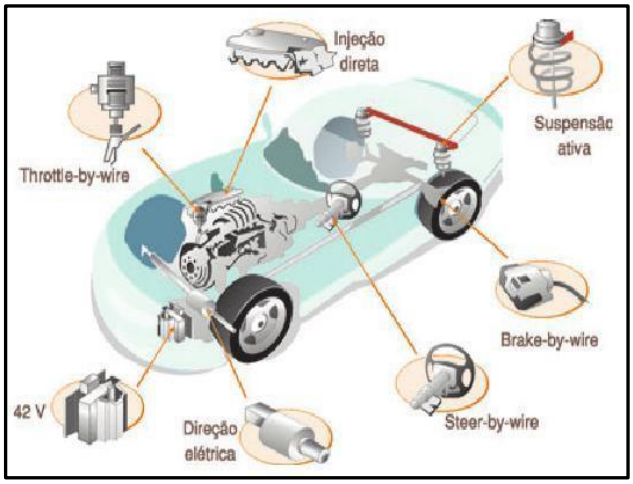
- Integração Física: Espaços inteligentes são definidos pela colaboração intensa entre elementos computacionais e componentes do mundo físico (Figura 4).

Figura 4 – Exemplo de Integração Física

Fonte: Cook et al. (2009).

- Invisibilidade: Quanto mais presente em nossa vida uma tecnologia estiver, menos perceptível ela deve ser. Em termos gerais, extingue-se a integração entre o homem e o computador e possibilita que as duas entidades convivam em perfeita simbiose. A ideia é que a computação seja imperceptível como os motores que hoje estão embarcados em diferentes equipamentos (Figura 5).

Figura 5 – Exemplo de Invisibilidade.



Fonte: Sabereletronica (2016).

- Pró-Atividade: E a capacidade do sistema se antecipar a intenção do usuário, por exemplo, um sistema que identifica que o carro está saindo da curva e, assim, aciona dispositivos de segurança, como afivelar o sinto de segurança para o motorista e o fechamento das janelas abertas (Figura 6).

Figura 6 – Exemplo de Pró-Atividade

Fonte: Caricos (2016).

- Sensibilidade ao Contexto: Sistemas Ubíquos precisam recorrer às informações de contexto para adaptar seu comportamento e funcionalidades. Essa característica permite que cada sistema conheça o ambiente em que está operando e se ajuste de acordo com o contexto sem que o usuário tenha conhecimento.

- Interoperabilidade Espontânea: É a possibilidade de integração dinâmica entre componentes móveis e a infraestrutura do sistema, sem a intervenção do usuário (Figura 7).

Figura 7 – Exemplo de Interoperabilidade Espontânea

Fonte: Megalindas (2016).

- Interfaces Naturais: Em ambientes inteligentes é preciso que se busque técnicas para que os recursos normalmente utilizados no dia a dia de uma sociedade, como gestos, voz e olhares, permaneçam como meio de comunicação entre homens e máquinas (Figura 8).

Figura 8 – Exemplo de Interação Natural

Fonte: Xbox (2016).

- Coordenação: ambientes ubíquos são altamente dinâmicos. Interações síncronas e assíncronas através de várias entidades computacionais podem ser realizadas a todo o momento. Estas interações precisam ser realizadas de um modo coordenado.
- Adaptação e Tolerância a falhas: Sistemas ubíquos devem considerar não somente mudanças de contexto, mas também falhas nos serviços disponíveis, na rede e dispositivos. Adaptar-se a essas falhas evita o mau funcionamento do sistema.

- Mobilidade: Usuários, dispositivos e serviços podem mover-se dentro de um mesmo ambiente e entre ambientes distintos. Desse modo, é necessário prover mecanismos para dinamicamente aprender sobre todos os serviços disponíveis nas proximidades do usuário.

2.2. INTERNET DAS COISAS

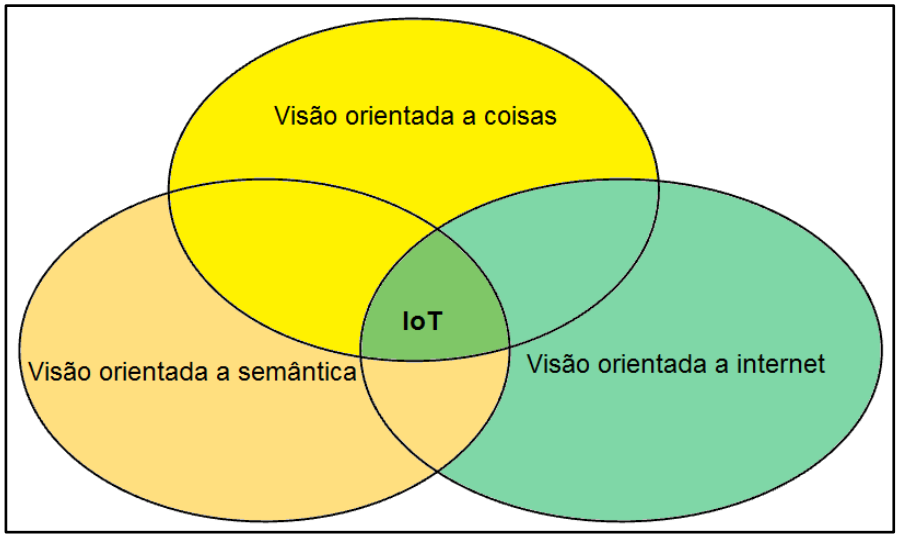
A Internet das Coisas ou IoT, na sigla em inglês, é uma expressão normalmente usada como sinônimo de ambientes conectados, web das coisas, internet do futuro e cidades inteligentes. Entrou em evidência a partir do ano 2000, sendo vinculada a

cenários que estão presentes no nosso dia a dia, tais como, carros, casas e cidades (Atzori et al., 2010).

A ideia de ubiquidade está presente na IoT, uma vez que a ubiquidade se refere à noção de algo que está presente em todos os lugares e em todos os momentos, persistente, sempre disponível e atuante (SANTAELLA, 2013).

A principal característica da IoT refere-se a objetos conectados em rede e que produzem ou processam informação em tempo real e de forma autônoma. Esta idéia é melhor compreendida como um paradigma computacional formado pela sobreposição de visões orientadas às coisas, à Internet e à semântica (ATZORi et al., 2010) (Figura 9).

Figura 9 - O paradigma da Internet das Coisas



Fonte: Atzori et al., (2010).

- A visão orientada às coisas prevê objetos cada vez mais inteligentes, apresentando competências como comportamento proativo, sensibilidade ao contexto e comunicação colaborativa.
- A visão orientada à Internet direciona para uma definição da rede, onde protocolos devem estar adaptados para permitir a troca de informações entre as coisas na IoT.

- A visão orientada a semântica propõe interconectar e organizar a informação gerada pela Internet das Coisas, para modelagem e extração do conhecimento dos dados.

Neste contexto, considera-se a IoT como um paradigma computacional com implicações profundas no relacionamento entre homens e objetos. Uma vez que a definição é bastante ampla, pois reúne diversos fatores como, por exemplo, a criação de uma rede global, padronização e identidade dos objetos a IoT pode ser delimitada como: conectar objetos dotados da capacidade de agirem por conta própria, com ou sem supervisão humana. Essas redes têm a característica de conectar não apenas humanos a objetos, mas também objetos a objetos e humanos a humanos. Assim, a IoT passa a ser capaz de controlar uma série de ações do dia a dia sem a necessidade que as pessoas estejam atentas e no comando (SINGER, 2012).

2.3. AMBIENTES INTELIGENTES

Para que a Computação Ubíqua e Pervasiva se concretize, será necessário o desenvolvimento de Ambientes Inteligentes (AmI) que possam integrar diferentes perfis de pessoas (ALENCAR ; NERIS, 2013), inclusive as com necessidades especiais, sendo a utilização de comandos de voz, gestos e ondas cerebrais, alternativas plausíveis de interação.

Neste contexto, para promover a participação social concreta das pessoas com necessidades especiais é necessário a utilização de novas tecnologias, o que requer novas discussões e análise das possíveis soluções disponibilizadas com o avanço da Computação Ubíqua e Pervasiva (ALENCAR ; NERIS, 2013).

Com o objetivo de contribuir com pesquisas nessa área, nesta dissertação é proposta a criação de uma plataforma para o desenvolvimento de aplicações Ubíquas em Ambientes Inteligentes, com a interação Humano-Computador via voz.

AmI tem sido caracterizada de diferentes maneiras, como pode ser observado na Tabela 1. Os principais atributos enfatizados pelos pesquisadores na área são: (S)

Sensível, (A) Ágil, (Adp) Adaptável, (T) Transparente, (O) Onipresente e (I) inteligente (Tabela 3).

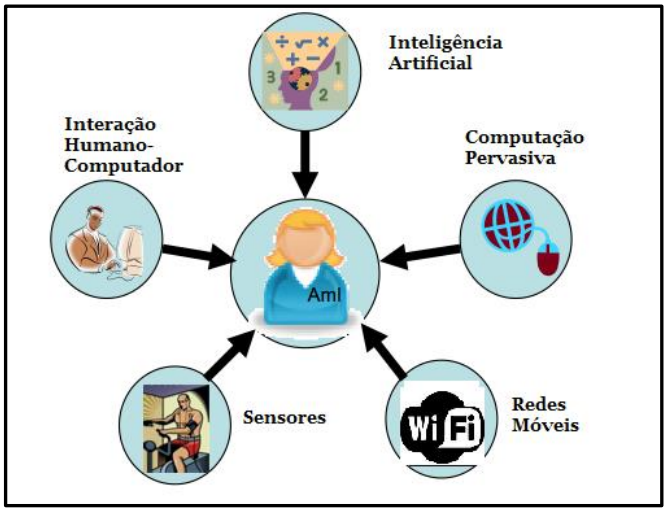
Tabela 3 – Definições de AmI

Definições de AmI	S	A	Adp	T	O	I
Tecnologia em desenvolvimento que faz cada vez mais os ambientes serem sensíveis e receptivos à presença humana (Aarts; Encarnacao, 2006).	X	X				
Um ambiente cercado por objetos inteligentes que vão reconhecer a presença de pessoas e responder a elas de maneira transparente (Ducatel et al., 2010)	X	X		X	X	
Ambientes inteligentes implica na utilização da inteligência de objetos em nossa volta (Maeda; Minami, 2006)					X	X
A tecnologia inteligente deve desaparecer em nosso ambiente para trazer aos seres humanos uma vida fácil e divertida (Crutzen, 2006)		X		X	X	
Em AmI as pessoas estarão cercadas por redes e dispositivos inteligentes, capazes de identificar as necessidades de seus usuários e antecipar ações para se adaptar às necessidades (Vasilakos; Pedrycz, 2006)	X		X	X	X	X

Fonte: Sharma et al. (2014).

A partir das definições e características resumidas na Tabela 3, pode ser observado que os aspectos da AmI incorporam a consciência do contexto, a onipresença e a ubiquidade. Como resultado, AmI incorpora pesquisas em várias esferas, abrangendo estudos nas áreas de Inteligência Artificial, Computação Pervasiva, Interação Humano-Computador, Redes Móveis e Sensores (COOK et al., 2009) (Figura 10).

Figura 10 – Áreas relacionadas a AmI



Fonte: Cook et al. (2009).

2.3.1. *Frameworks* para ambientes inteligentes

Os *frameworks* para aplicações ubíquas utilizam em geral conceitos de Inteligência Ambiental (AUGUSTO; MCCULLAGH, 2007), de Computação Sensível ao Contexto (DEY et al., 2001) e de Prototipação de Aplicações Ubíquas (WEIS et al., 2007). Segundo Mareli et al. (2013) a Inteligência Ambiental define o AmI, que é o espaço onde estas aplicações funcionam. O comportamento de sistemas ubíquos é definido a partir de técnicas aplicadas na Computação Sensível ao Contexto. A prototipagem é utilizada para manipular e testar o funcionamento de aplicações no ambiente.

Os sistemas com enfoque na Computação Ubíqua aplicada no AmI são baseados geralmente em uma arquitetura em camadas. Na camada inferior encontra-se o espaço físico com seus ocupantes, e em uma camada acima estão os sensores coletando informações de contexto e os atuadores provendo serviços para atender às necessidades destes ocupantes. Uma camada intermediaria (*middleware*) é definida entre as tomadas de decisão e as interações com o ambiente, incluindo também conceitos e mecanismos para a construção de software dessa classe de sistemas. As decisões podem ser tomadas por um ocupante ou através de mecanismos de inteligência artificial. Ha propostas na literatura que caracterizam este tipo de ambiente, dentre eles o termo “*smart home*” (casa inteligente) se mostra como um dos mais conhecidos (HELAL et al., 2005).

2.4. INTERAÇÕES NATURAIS

O termo interação natural (do inglês NUI - Natural User Interfaces) nasceu em um momento tecnológico em que a Interação Homem-Computador (IHC), se torna cada vez mais cômodo ao usuário. Dentro de um contexto de computação ubíqua, a interação deve ocorrer de forma transparente, pois para os usuários é inviável o aprendizado específico de diferentes tipos de interfaces. Desta forma, sugere uma inversão de papéis, na qual ao invés das pessoas interpretarem a aplicação em questão, esta passa a entender o individuo, percebendo suas intenções e usando-as para alcançar uma interação bem-sucedida. Este, aliado a outros conceitos como liberdade, uso de metáforas e *feedback* em tempo real, dão origem ao campo de pesquisa chamado interação natural (VALLI, 2007).

As pessoas se comunicam normalmente através de gestos, expressões faciais, movimentos corporais e da fala. Da mesma forma, percebem o mundo a sua volta, olhando, ouvindo e manipulando materiais físicos (FIGUEIREDO et al., 2012). Segundo os autores, estes conceitos caracterizam a interação natural.

A NUI oferece maneiras ricas para interagir com o mundo digital, inovando com a utilização de recursos humanos existentes. Eles incluem e muitas vezes combinam diferentes modalidades de entrada, tais como, voz, gesto, o olhar, as interações do corpo, toque e interações sem toque (VETERE et al., 2014). Este paradigma de interação, deve ser desenvolvido de forma a permitir que pessoas ajam e se comuniquem de maneira que elas estão predisposta naturalmente (O'HARA et al., 2013).

Neste contexto, as aplicações passam a entender o individuo, percebendo suas intenções e usando-as para gerar ações bem sucedida, ao invés das pessoas interpretarem a aplicação em questão. Ou seja, dentro de um contexto de computação ubíqua, a interação deve ocorrer de forma transparente ao usuário. A tendência atual é ter interfaces cujas implementações se utilizem progressivamente da naturalidade, tendo o homem como ponto inicial, resultando em um dispositivo que tenha uma integração cada vez melhor com as pessoas.

2.5. MODALIDADE DE INTERAÇÃO

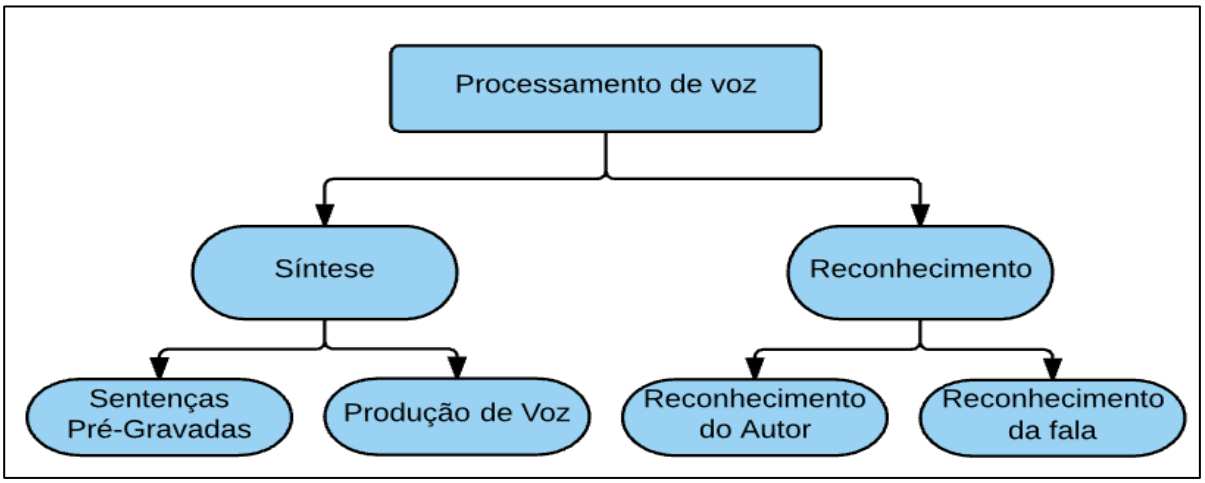
A área de IHC vem se remodelando, visando obter melhores resultados para as necessidades e satisfação das pessoas. Desta forma, com o advento da IoT, é notória a evolução de AmIs que tem como base as interações naturais, este fato possibilita aumentar a sensação de imersão e melhora da experiência dos usuários (FIGUEIREDO et al., 2012).

A voz pode ser apontada como uma das formas mais naturais de interação entre pessoas. Desde a década de 50, estudos em Inteligência Artificial vislumbram o uso da voz como ferramenta para a interação entre máquinas e pessoas, mas as limitações de hardware e software foram impactantes (MARTINS; BRASILIANO, 2012).

Os autores ainda informam que o uso das interfaces não convencionais – tais como Realidade Virtual, Realidade Aumentada e Interfaces Baseadas em Voz – surge como uma forma de tentar tornar estas aplicações mais naturais e mais fáceis de serem usadas, gerando maior satisfação aos usuários não especialistas.

As tecnologias da fala, usada em interfaces baseadas em voz, destinam-se a facilitar a interação entre o utilizador e as máquinas, complementando ou substituindo, por exemplo, o teclado e o mouse, como também, um determinado periférico elétrico ou eletrônico. Tradicionalmente, são duas áreas incluídas no processamento da fala: (i) a síntese da fala, sistema que permite a conversão de texto em fala; e (ii) o reconhecimento automático da fala, sistema que possibilita a conversão de voz em texto (LISTERRI ; MARTÍ, 2002). Esta ideia é ilustrada na Figura 11.

Figura 11 - Processamento de voz



Fonte: Autor deste trabalho (2016).

2.6. LÍNGUA, FALA E VOZ

A língua, no seu sentido físico, é um órgão muscular que desempenha um papel significativo na produção e articulação dos sons da fala. Entretanto, no sentido filológico, a língua é considerada um sistema gramatical que pertence a um grupo de indivíduos. Desta forma, possibilita estabelecer a comunicação entre o emissor e o receptor, através de um código. Pode-se dizer que ela é um instrumento primordial do entendimento do mundo, uma criação da sociedade, porém não é imutável; ao contrário, tem de viver em perpétua evolução, paralela ao organismo social que a criou. Ou seja, todas as línguas evoluem com o tempo, de acordo com a evolução da sociedade (NASCIMENTO, 2011).

Além de evoluir com o tempo, a língua, apresenta variação consoante, na sua utilização e intenção discursiva. Essas variações ocorrem em todos os níveis de uma língua: fonético, fonológico, morfológico, sintático e estilístico (BRUSTOLIN, 2009). Sendo assim, é inseparável da cultura do local onde se fala, o que significa que é acompanhada de entonações, gestos, olhares e expressões faciais, entre outros tipos de linguagem (SANTOS, 2013).

O Dicionário On-line de Língua Portuguesa, Michaelis, define linguagem como “conjunto de sinais falados, escritos ou gesticulados de que se serve o homem para exprimir esses pensamentos e sentimentos.” (DICIONÁRIO..., 2016). Ou seja, a linguagem falada só pode existir e manifestar-se a partir do momento em que um

indivíduo aprende uma língua e será concretizada através do discurso, de forma que restringimos a uma atividade coletiva realizada por meio de um código formado por palavras regidas por leis combinatórias às quais pertencem a um grupo específico, como, por exemplo, a língua brasileira.

A fala é a língua no momento em que está sendo empregada por uma pessoa na utilização de palavras e regras gramaticais dessa língua. Este processo é estritamente individual, uma vez que cada pessoa tem a sua forma própria de se manifestar no meio social. Esta, ao usufruir dos seus conhecimentos linguísticos, o torna apto a expressar seu pensamento de acordo com sua visão de mundo obtida de acordo com a sua experiência. Já a voz, é o som produzido pelo aparelho fonador humano (NASCIMENTO, 2011).

2.6.1. Síntese de Voz

A geração automática pelo computador em forma de onda de voz, conhecida como síntese de voz, é definida como o processo de produção artificial de voz humana em um sistema computacional. Dessa forma, é considerado sintetizador de voz, podendo ser desenvolvido em software ou hardware. Um sistema capaz de converter texto para fala - TPF ou mais comumente chamado de TTS, do termo em inglês: "*Text-to-Speech*" – faz a transformação de texto em voz, muitas vezes, denominada transcrição fonética, sendo sua qualidade definida pelo grau de similaridade com a voz humana (RODRIGUES et al. 2012).

Em resumo, o objetivo principal dos sistemas TTS's é reproduzir a fala de um ser humano a partir de um texto de entrada, em linguagem natural. Os primórdios desse tipo de síntese de fala mostram o que foi desenvolvido mecanicamente, até que, com a evolução tecnológica, se fosse capaz de utilizar um computador capaz de processar e sintetizar voz.

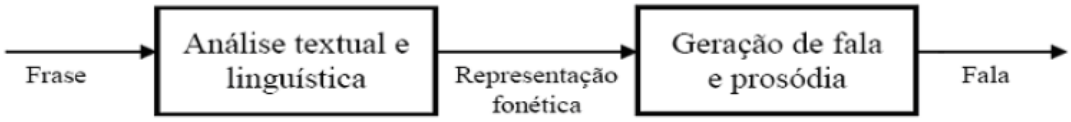
2.6.2. Sistemas de Conversão Texto-Fala

Jufarsky e Martin (2009) afirmam que um sistema de conversão texto-fala é composto por dois módulos claramente distintos que requerem para sua realização uma metodologia e conhecimento de base radicalmente distinto: análise linguística e

geração da fala. Desta forma, o funcionamento de um sistema TTS pode ser dividido em duas fases principais. A primeira fase consiste na análise do texto, na qual o texto de entrada é transcrito para uma representação fonética, e a segunda fase incide na geração de onda de voz, onde a saída acústica é produzida a partir dessa informação fonética.

A Figura 12 ilustra as fases da síntese de voz onde, basicamente, uma cadeia de caracteres é processada e analisada de acordo com a sua representação fonética com algumas informações adicionais como, entonação e duração. A última etapa para a saída da fala é a sintetização na forma de onda (GOMES, 2007).

Figura 12 - Segmentação da Síntese da fala



Fonte: Autor deste trabalho (2016).

2.6.3. Aplicações para TTS

O conceito de síntese (TTS) apareceu em meados dos anos oitenta, como resultado de importantes desenvolvimentos na síntese de voz e técnicas de processamento de linguagem natural. Desde então, a utilização da voz como forma de interação tem crescido exponencialmente. Existem inúmeras aplicações com potencial para usufruir desta forma de interação, sendo uma das suas vantagens a possibilidade de ser utilizada em sistemas para pequenos dispositivos, tais como, *smartphones* e *tablets*, como em sistemas sofisticados, como plantas industriais.

Os sistemas de conversão texto-fala encontram-se atualmente em uma série de aplicações de grande utilidade, nomeadamente, como elementos de uma interface por voz para computador, assegurando a função de saída acústica para avisos ou informações ao utilizador dos conteúdos das mensagens que o sistema procurar transmitir-lhe. Este tipo de interface pode ser aplicado em quaisquer sistemas de informação, tais como, estações de transporte e sistemas comerciais, principalmente

como forma de prover a inclusão digital de pessoas cegos, amblíopes, idosos e iletrados.

Segundo Dutoit (1997) existem muitas outras aplicações em potenciais que se destacam em sistemas TTS como, por exemplo:

- Serviços de Telecomunicações: Sistemas TTS tornam possível acessar informação textual por meio do telefone. Sabendo que cerca de 70% das chamadas de telefone atualmente requerem muita pouca interatividade, um prospecto é digno de ser considerado. Textos podem abranger desde pequenas mensagens como, eventos culturais locais (cinemas, teatros) até enormes bases de dados que dificilmente poderiam ser lidas e armazenadas como a fala digitalizada.

- Ensino de Linguagens: Sintetizadores TTS de alta qualidade podem, com ajuda de um sistema de aprendizagem, promover ferramentas de ensino a novas linguagens. Esse tipo de projeto tem amplo mercado, no entanto, ainda não está plenamente implementado devido à necessidade de aperfeiçoamento dos sistemas síntese de voz.

- Ajuda a Pessoas Deficientes: Para ajudar pessoas que não podem pronunciar palavras, os sistemas computacionais modernos podem, a partir de outros mecanismos de entrada de dados, montar sentenças de voz sintética em poucos segundos.

- Livros e brinquedos falantes: O mercado de brinquedos tem se aproximado cada vez mais aos recursos de síntese de voz. Muitos brinquedos falantes têm sido criados, mas as limitações de qualidade invariavelmente interferem na ambição educacional dos produtos. Sintetizadores de alta qualidade melhoram essa situação, mas são bastante caros para se agregar ao valor dos brinquedos.

- Monitoramento Vocal: Em alguns casos, informação oral é mais eficiente do que mensagens escritas. O apelo é mais forte para cenários aonde a atenção do usuário pode estar focada em outras fontes visuais de informação. Consequentemente, a idéia de incorporar sintetizadores de voz no

gerenciamento ou controle de sistemas como, em cabines de aviões, para gerar alerta aos pilotos, é uma boa alternativa para diminuir a carga cognitiva de cenários que geram muitas informações simultâneas.

No próximo capítulo será ampliada a discussão a respeito da voz como forma de interação, abordando principalmente as tecnologias existentes para a interpretação de comandos de voz.

3 RECONHECIMENTO AUTOMÁTICO DA FALA

O objetivo deste capítulo é contextualizar a voz como meio de interação e a importância de uma abordagem eficaz para o seu desenvolvimento. Para isso, será discutido a finalidade do reconhecimento automático da fala e algumas tecnologias que auxiliam no seu desenvolvimento, bem como as principais dificuldades enfrentadas em seu processo.

3.1. INTERFACE NATURAL DE USUÁRIO

A principal vantagem do reconhecimento de voz é permitir ao usuário a interação de forma mais natural. Para Wigdor e Wixon (2011) Interface Natural de Usuário (*Natural User Interface* - NUI) é uma modalidade de interação que faz o usuário sentir e agir naturalmente, como se estivesse interagindo com outra pessoa. Blake (2011) acredita que são interfaces projetadas a fim de reutilizar habilidades já existentes do usuário para interação direta com o conteúdo. Segundo Liu (2010), a NUI, baseada em voz torna a experiência no processo de comunicação com o usuário mais natural e confiável, sendo reconhecida como um canal conveniente e eficaz para o compartilhamento de informações.

Interfaces Naturais estão cada vez mais presentes no dia a dia das pessoas, pois proporcionam uma experiência mais próxima das interações humanas nas relações sociais e com o meio ambiente, tornando quase transparente a interface entre homem e máquina (BLAKE, 2011).

Na medida em que a tecnologia avançar e a produção de hardware e software acompanhar este avanço de modo que seja viável comercialmente, certamente trará para o cotidiano das pessoas novas experiências de interação, cada vez mais intuitivas e naturais.

Para a evolução na área de interações naturais é necessário conhecer o perfil dos usuários, as tarefas que devem ser realizadas e o contexto em que a aplicação será utilizada para que a interface seja capaz de extrair as habilidades humanas necessárias para que a experiência do usuário seja a mais gratificante possível.

3.2. RECONHECIMENTO DE VOZ

Na abordagem ASR, que em português significa “Reconhecimento Automático da Fala”, a voz é capturada em forma de sinal digital e convertida para textos escritos. O sistema ASR tem sido estudado desde 1950. Inicialmente, foi desenvolvido o primeiro reconhecedor de dígitos isolados com suporte a apenas um locutor. Na mesma década, foi introduzido o conceito de redes neurais, mas devido a muitos problemas práticos a ideia não foi seguida no âmbito da ASR (OLSON ; BELAR, 1956).

Com o passar dos anos, as muitas limitações técnicas foram sendo superadas, além da globalização e popularização dos computadores, o que elevou o número de pesquisas na área de processamento de voz (LECOUTEUX et al., 2011).

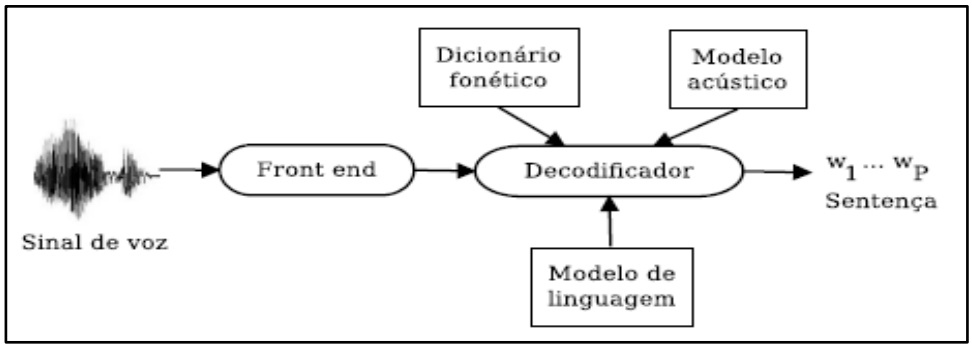
Na década de 80, os sistemas ASR, tentavam aplicar um conjunto de regras gramaticais e sintáticas à fala. Caso as palavras ditas caíssem dentro de um conjunto de regras, o programa poderia determinar quais eram aquelas palavras. Para isso, era preciso falar cada palavra separadamente (sistemas de palavras isoladas), com uma pequena pausa entre elas. Apesar dos avanços, muitos problemas ainda precisavam ser resolvidos, tais como, sotaques, dialetos e outras características inerentes à língua, o que dificultou a disseminação dos sistemas baseados em regras. Foi então que os pesquisadores iniciaram estudos para reconhecimento de palavras conectadas utilizando métodos estatísticos, com maior destaque para os modelos ocultos de Markov (HMM) (FERGUSON, 1980).

Um sistema ASR é, tipicamente, composto por cinco blocos: front-end, dicionário fonético, modelo acústico, modelo de linguagem e decodificador, conforme demostrado na Figura 13.

O desenvolvimento de interfaces homem-máquina controladas pela voz visa substituir, em certas aplicações, as interfaces tradicionais, tais como, teclados, painéis e dispositivos similares. Neste cenário, se insere o reconhecimento de voz

como alternativa para uma interface mais natural entre os sistemas computacionais e o homem.

Figura 13 - Principais blocos de um sistema de reconhecimento de voz



Fonte: Autor deste trabalho (2016).

Nos dias atuais existem dois tipos básicos de sistemas de reconhecimento de voz RABINER (1993):

- Reconhecedor de palavras isoladas – É o tipo de sistema utilizado para a interpretação de vocabulários pequenos e em ambiente livres de ruído. Existem diversos sistemas comerciais amplamente utilizados, cuja taxa de reconhecimento, independente do locutor, é de aproximadamente 100% para vocabulários com até 10 palavras e de 95%, para vocabulários até 1000 palavras. A principal exigência dos reconhecedores de palavras isoladas é que o locutor deve proferir as palavras com pausas de aproximadamente 200ms. Apesar disso, é bastante empregada na área automobilística.

- Reconhecedor de voz contínua – É um sistema mais complexo e difícil de ser implementado, pois devem ser capazes de lidar com todas as características e vícios da forma natural de falar.

Embora muito se tenha aprendido a respeito da forma como implementar sistemas práticos e úteis de reconhecimento de voz, ainda resta um árduo caminho a ser percorrido. Cinco são os principais fatores que determinam a complexidade de qualquer sistema de reconhecimento de voz (POZA, 1991):

- O locutor – É o agente que introduz maior variabilidade na forma de onda do sinal de entrada requerendo, portanto, que o sistema de reconhecimento seja altamente robusto. Uma pessoa não pronuncia uma locução sempre da

mesma forma devido a distintas situações físicas e psicológicas (chamadas variações interlocutores). Existem, ainda, diferenças entre os tipos de locutores (homens, mulheres, crianças e idosos) e o perfil de cada um.

- A forma de falar – Este fator determina a complexidade do sistema. Os seres humanos pronunciam as palavras de forma contínua e devido à inércia dos órgãos articulatórios, que não se movem instantaneamente, são produzidos os efeitos coarticulatórios. Estes, unidos às variações introduzidas pela prosódia (pronúncia regular das palavras em harmonia com a acentuação), fazem com que haja diferença entre uma mesma palavra dita no início e no meio de uma frase.

- O vocabulário – Corresponde às diferentes palavras que o sistema deve ser capaz de reconhecer. Portanto, quanto maior é o vocabulário mais árdua torna-se a tarefa de reconhecimento, por dois motivos: (i) porque ao aumentar o número de palavras é mais provável que surjam palavras parecidas entre si; e (ii) porque o tempo de tratamento é proporcionalmente maior à medida que aumenta o número de palavras a serem comparadas.

- A gramática – Conjunto de regras que limita o número de combinações permitidas às palavras do vocabulário. Em geral, a existência de uma gramática em um reconhecedor ajuda a melhorar a taxa de reconhecimento, ao eliminar ambiguidades. Além disso, contribui para diminuir a carga computacional, ao limitar o número de palavras em uma determinada frase a ser interpretada.
- O ambiente - Responsável pela inserção de ruído. Trabalhando-se fora do ambiente de laboratório tal influência torna-se inevitável, pois este ruído pode aparecer de diversas formas, desde vozes de outros locutores, sons de equipamentos e provocados pelo próprio locutor, tais como: tosses, espirros, estalo dos lábios, suspiro, respiração forte, entre outros.

As ferramentas de reconhecimento de voz estão definidas em estruturas locais e em nuvem, cada uma delas com características específicas. Um estudo comparativo realizado por Morbini et al. (2013) apontou as seguintes características:

- Estrutura em nuvem - Servidores em nuvem apresentam uma menor taxa de erro por palavra interpretada. Este fato se confirma por causa do processamento distribuído, os quais possuem alto poder de desempenho alinhado a uma vasta base de dados para o treinamento de algoritmo. Contudo, os atrasos inerentes a rede de dados compromete as respostas do sistema, podendo inviabilizar parte do processo.

- Estrutura local - Estruturas locais tendem a ter um desempenho melhor por não terem dependência da Internet. Desta forma, o tempo de retorno de uma solicitação é mais aceitável, principalmente se o sistema for composto por um vocábulo de reconhecimento simples e limitado. Caso a gramática seja extensa, o processamento e a utilização da memória aumenta, obrigando o sistema a dispor de um bom conjunto de hardware para a interpretação das palavras.

3.3. TÉCNICAS DE TRADUÇÃO DE VOZ

3.3.1. Redes Neurais Artificiais (Artificial Neural Networks - ANN)

Um dos ramos da inteligência artificial utilizados nos algoritmos para a interpretação de comandos de voz são as Redes Neurais Artificiais (RNAs). Souza (1999) atribui seu uso crescente devido à sua capacidade de fazer suposições mais delicadas a respeito da distribuição dos dados de entrada do que métodos estatísticos tradicionais e à capacidade de auxiliar na resolução de problemas de natureza não-linear, cujo funcionamento é inspirado no próprio cérebro humano.

Segundo Haykin (1999), uma rede neural é um processador simples, que têm a capacidade de armazenar conhecimento experimental e torná-lo disponível para uso. Ela se assemelha ao cérebro em dois aspectos:

- O conhecimento é adquirido pela rede, a partir de seu ambiente, através de um processo de aprendizagem.

- Forças de conexão entre neurônios, conhecidos como pesos sinápticos, são utilizados para armazenar o conhecimento adquirido.

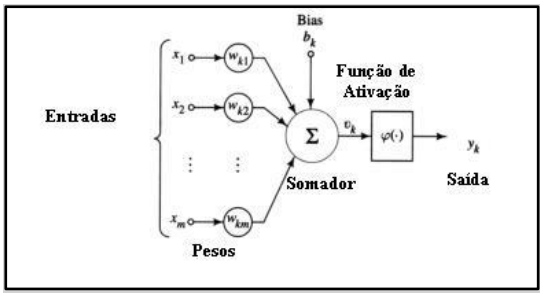
O procedimento pelo qual uma RNA encontra as soluções passa por um processo de aprendizado, onde uma série de amostras de entrada e saída são apresentadas às suas unidades elementares que, por si só, encontram as características necessárias para representar a informação fornecida e posteriormente, definir o sistema resultante (SOUZA, 1999).

As RNAs têm a capacidade de aprender com os exemplos que lhe são apresentados e generalizar a informação aprendida, sendo possível, classificar as amostras de dados desconhecidos, mas que se assemelham as informações contidas na etapa de treinamento. As RNAs são capazes de extrair características que não estejam explicitamente apresentadas sob a forma de exemplos (ou amostras de entrada).

O cérebro é destinado a cuidar em nosso corpo no que se trata de emoção, raciocínio e funções motoras. As RNAs, por sua vez, têm como ambição simular este mundo de atividades realizadas pelo cérebro, implementando o seu comportamento básico e sua dinâmica.

O modelo matemático de um neurônio artificial foi proposto por Warren McCulloch, psiquiatra e neuroanatomista, e Walter Pitts, matemático, em 1943. O modelo em si era uma simplificação do neurônio biológico até então conhecido na época. Para representar os dendritos, o modelo constou de *n* terminais de entrada de informações *x1, x2, ..., xn* e simplesmente um terminal de saída *y*, para representar o axônio. Cada entrada apresenta um coeficiente ponderador que visa à simulação das sinapses, sendo que estes coeficientes são valores reais. De forma análoga ao neurônio biológico, a sinapse só ocorre quando a soma ponderada dos sinais de entrada ultrapassa um limiar pré-definido, realizando, portanto, uma atividade semelhante à do corpo humano. No modelo proposto, o limiar foi definido de forma Booleana (dispara ou não dispara), resultante de uma função de ativação, conforme pode ser visto na Figura 14 a seguir.

Figura 14 - Neurônio de McCulloch e Pitts



Fonte: Adaptado de Medeiros (2006).

A saída *y* do neurônio de McCulloch e Pitts pode ser equacionada por:

$$net_i(t) = \sum_{j=1}^{n} w_{ij} x_j(t)$$

O somatório de todas estas entradas, multiplicadas por suas respectivas forças de conexão sináptica (os pesos), dá origem ao chamado "net" de um neurônio. Desta forma, as redes neurais são boas para tarefas que exijam: (i) reconhecimento, classificação e a associação de padrões; (ii) resistência a ruído; (iii) robótica; e (iv) processamento de sinais e imagem.

O sistema é altamente integrado, trabalha de modo paralelo e distribuído com vários dados interconectados, constituídos de uma alta capacidade de processamento que processa vários algoritmos matemáticos, a fim de gravar conhecimento e utilizá-lo. Entre as principais características: (i) capacidade de aprender por meio de exemplos e de generalização; (ii) tolerância a falhas; (iii) possuir alta capacidade de adaptação; e (iv) robustez diante de informações falsas (HAYKIN, 1999).

3.3.2. Hidden Markov Model (HHM)

A síntese de voz baseada em HMM vem se tornando popular devido à sua flexibilidade em sintetizar voz considerando características como estilo e individualidade da fala do locutor, além da possibilidade de expressar aspectos emocionais na voz sem a necessidade de uma grande quantidade de amostras dos dados (RABINER, 1989).

O Modelo de Markov escondido (*Hidden Markov Model* – HMM) é um método estatístico de reconhecimento de voz, sendo um dos mais utilizados devido à eficiência que apresenta entre a carga computacional e sua flexibilidade.

Define-se como HMM, um processo de Markov que possui um número contável de estados, no qual uma dada observação não é estado e sim uma função probabilística deste. Em outras palavras, o HMM é um processo estocástico que só pode ser observado através de um outro conjunto de processos estocásticos que produzem a sequência de observações; daí o nome hidden (oculto, escondido).

O HMM para o reconhecimento da fala tem demonstrado muita eficiência na caracterização das propriedades temporais e espectrais do sinal de voz e seu uso é baseado nas seguintes assertivas:

1. A fala pode ser segmentada, dividida em estados, nos quais a forma de onda do sinal de voz pode ser considerada estacionária. Assume-se que a transição entre tais estados seja instantânea.

2. A probabilidade de uma certa “observação” ser gerada depende apenas do estado atual e de nenhum símbolo gerado anteriormente.

É possível usar o HMM para representar qualquer unidade da fala. Para sistemas de reconhecimento de vocabulários pequenos, normalmente, utiliza-se HMM para modelar diretamente as palavras, enquanto que para grandes vocabulários é utilizado para modelar sub-palavras.

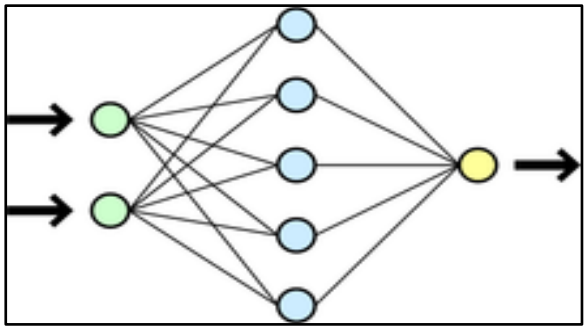
3.3.3. Multi Layer Perceptron (MLP)

Conforme dito anteriormente, as RNAs com uma camada de neurônio são capazes de resolver problemas linearmente separáveis, contudo apesar de resolver uma gama vasta de problemas, existe, por outro lado, uma outra vasta coleção de problemas não linearmente separáveis. Este problema foi proposto por Minsky e Papert na década de 70, quando, em suas publicações, depreciaram a habilidade das RNAs de encontrar soluções para simples problemas como, por exemplo, a

modelagem do “Ou Exclusivo” da lógica digital. A solução encontrada para contornar este problema e como consequência retomar as pesquisas sobre RNA, até então desacreditas por Minsky e Papert, foram as estruturas neurais de múltiplas camadas, também conhecida como redes Multi Layer Perceptron (MLP) (CYBENKO, 1989).

As redes MLP apresentam a arquitetura mostrada na Figura 15, onde se encontram a camada de entrada, as camadas intermediárias (ou ocultas) e a camada de saída. O número de variáveis da camada de entrada depende diretamente do número das características agrupadas no vetor das amostras. O número de neurônios das camadas intermediárias depende da complexidade do problema. E a camada de saída contém o número de neurônios necessário para executar a codificação das amostras de entrada.

Figura 15 - Arquitetura das Redes MLP



Fonte: Autor deste trabalho (2016).

O número de neurônios das camadas intermediárias é determinado de forma empírica, atentando para o caso de *overfitting* (ou superajuste), que é o caso onde existe uma grande quantidade de neurônios e a estrutura em vez de generalizar as informações, acaba por memorizar os padrões apresentados, não sendo capaz de classificar padrões semelhantes (HAYKIN, 1999).

Outro efeito do superajuste é que a RNA além de armazenar as características relevantes extraídas das amostras, guarda em seus pesos informações de ruídos que a princípio não revelam interesse. Por outro lado, caso o número de neurônios seja inferior ao desejado, pode ocorrer um *underfitting* (ou baixo ajuste), e a RNA

não converge para uma resposta devido a uma sobrecarga de informações a serem armazenadas em poucos pesos.

Flanagan (1972) define o funcionamento de uma rede neural MLP em três estágios: (i) a entrada de um conjunto de informações (vetor de características extraídas da amostra de voz a ser analisada); (ii) o cálculo das saídas da rede; e (iii) a classificação do sinal de entrada. Assim, o reconhecimento de cada palavra é realizado executando-se essa rotina a partir do vetor de características extraídas do sinal de entrada.

3.4. TECNOLOGIA DE RECONHECIMENTO LOCAL

3.4.1. Julius

Julius é um software decodificador de alto desempenho para pesquisadores e desenvolvedores de sistemas com reconhecimento de voz. Realiza decodificação quase que em tempo real, na maioria dos computadores, utilizando o modelo oculto de Markov. Possui um formato padrão para lidar com outros conjuntos de ferramentas de modelagens como, por exemplo, HTK, CMU-Cam SLM kit de ferramentas. Os sistemas operacionais suportados são Linux e Windows. A distribuição é com licença aberta, associado ao código fonte e tem sido usado por muitos pesquisadores e desenvolvedores (GAIDA et al., 2014).

Silva et al. (2010) descrevem o Julius como sendo um sistema de reconhecimento de fala para o português brasileiro, apropriado para o reconhecimento de vocabulário extenso e funcionamento em tempo real. O sistema apresenta uma interface simples de programação para facilitar a tarefa de desenvolvedores voltados para a área de interações naturais via fala.

Para Oliveira et al. (2012) o fato do Julius ser uma ferramenta independente de idioma, desde que seja fornecido o dicionário apropriado, o modelo de linguagem e o modelo acústico, é uma das principais motivações para a sua adoção.

O Julius está sendo empregado em diversas aplicações (Neto et al., 2011), entre elas pode se destacar a utilização no desenvolvimento de um sistema multimídia

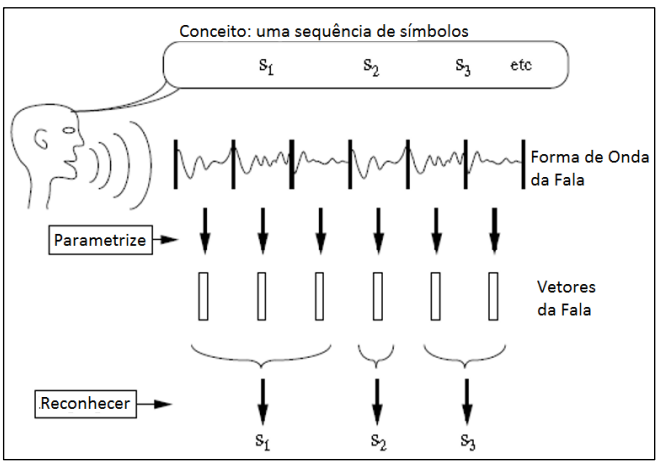
para apoiar na análise da fala de vídeo e programas de televisão. Segundo Pereira et al (2012), o Julius recebe os arquivos de áudio junto com a configuração que indica os recursos do idioma para o qual será realizada a tradução.

3.4.2. HTK - Hidden Markov Model Toolkit

O HTK é um conjunto de ferramentas portáteis para a construção e manipulação de modelos ocultos de Markov. É usado principalmente para a pesquisa de reconhecimento de fala, embora tenha sido utilizada para inúmeras outras aplicações, incluindo a investigação sobre a síntese de voz e reconhecimento de caracteres. Foi originalmente desenvolvido no Laboratório de Máquinas Inteligentes do Departamento de Engenharia da Universidade de Cambridge e atualmente está sob o domínio da Microsoft.

O software suporta modelos ocultos de Markov utilizando tanto distribuição contínua quanto discreta e pode ser usado para construir sistemas de modelos complexos. Como demonstrado na Figura 16, existem duas grandes etapas de processamento envolvidas para o reconhecimento. Em primeiro lugar, as ferramentas de treinamento HTK são usados para estimar os parâmetros de um conjunto de HMMs utilizando expressões de treino e as suas transcrições associadas. Em segundo lugar, os enunciados desconhecidos são transcritos usando as ferramentas de reconhecimento de HTK (YOUNG et al., 2002).

Figura 16 – Codificando/Decodificando HTK



Fonte: The HTK Book

3.4.3. Pocketsphinx

PocketSphinx é uma versão do Sphinx orientado para os sistemas embarcados e móveis, possibilitando a sua utilização em computadores portáteis e telefones celulares. O sistema de reconhecimento de voz tem a capacidade de tradução em tempo real. De acordo com seus criadores, é o primeiro sistema que consegue realizar traduções *off-line* rápidas e confiáveis e manter uma boa precisão. Possui o código aberto e treinável para reconhecer diferentes idiomas e dialetos, entretanto, vêm com um modelo treinado para Inglês Americano (DAINES et al., 2006).

Morbini et al. (2013), classificam o Pocketsphinx como uma versão da CMU Sphinx ASR otimizada para rodar em sistemas incorporados. É rápido, executado localmente e requer poucos recursos computacionais para a sua execução. Possui o reconhecimento de voz personalizável, no entanto, requer que sejam fornecidos os modelos de linguagem adequados para as aplicações. Estes modelos determinam qual linguagem será reconhecida pela tecnologia.

3.4.4. Sphinx

CMU Sphinx, também chamado Sphinx, foi desenvolvido na Universidade Carnegie Mellon por Kai-Fu Lee, se consagrando como um sistema eficiente para o reconhecimento de voz. As ferramentas são projetadas especificamente para plataformas de baixos recursos, com design flexível e suporte para vários idiomas como, por exemplo, o inglês, francês, mandarim, alemão, holandês, russo e tem capacidade de reconhecer modelos de outros idiomas.

O Sphinx é uma ferramenta de código aberto baseado no modelo discreto oculto de Markov (HMM), com parâmetros LPC-derivado, para reconhecimento de até 997 palavras com precisão entre 71% a 96% (GAIDA et al., 2014).

Kumar et al. (2011) avaliaram a precisão e a velocidade do Sphinx em relação ao PocketSphinx e constataram que em uma tarefa com vocabulário pequeno o PocketSphinx supera o Sphinx sobre a exatidão e a velocidade. No entanto, quando

a complexidade dos modelos acústicos e de linguagem aumentam, a precisão do Sphinx é melhor que o PocketSphinx. Sendo assim, o PocketSphinx é superior ao usar pequenos modelos acústicos e realizar reconhecimento em tempo real, entretanto para tarefas que permitem atrasos maiores em troca de uma melhor precisão, o Sphinx é a melhor escolha.

3.4.5. Microsoft Speech API

A Speech API fornece uma interface de alto nível, implementa todos os detalhes de baixo nível para controlar e gerenciar as operações em tempo real. A interface de programação reduz drasticamente a sobrecarga de código necessário para um aplicativo que utiliza reconhecimento de voz, tornando a tecnologia de voz mais acessível e robusta para uma ampla gama de aplicações. A Microsoft Speech SDK adiciona suporte a automação para desenvolver aplicações de voz com o Visual ® Basic, ECMAScript e outras linguagens de automação (GAO et al., 2007).

Segundo a Microsoft (2012) a interface de programação do Speech API reduz de forma substancial o código requerido para que se possa utilizar em um projeto o reconhecimento de voz e a transformação de textos em fala. Existem dois tipos básicos de máquinas Speech API: o sistema de Texto para Fala (TF) e o Reconhecimento de Voz (RV). O sistema TF sintetiza textos para serem ouvidos usando vozes sintéticas. Já o RV, faz o processo contrário, converte a voz humana em texto.

3.5. TECNOLOGIA DE RECONHECIMENTO EM NUVEM

3.5.1. Google Speech API

Google Speech API é uma ferramenta que permite que você controle seu computador usando comandos de voz por meio da API de reconhecimento de voz do Google. Em 2011, o Google implementou um recurso de HTML que reconhecia a fala do usuário, permitindo que ele o utilizasse ao fazer buscas no google.com, clicando no ícone do microfone. Em 2013 foi criado um novo recurso, o Speech API. O reconhecimento de voz faz a transcrição enquanto o usuário fala, não sendo

necessário dizer tudo de uma vez. Ele também reconhece comandos como “ponto”, “vírgula” e “novo parágrafo”.

Os principais objetivos do API incluem: (i) prover suporte a um sintetizador e reconhecedor de voz; (ii) disponibilizar uma interface multiplataforma robusta; (iii) permitir acesso ao estado da arte em tecnologia de voz; (iv) fornecer suporte à integração com outras funcionalidades; e (v) ser simples, compacta e fácil de aprender (SCHLOGL et al., 2013).

Morbini et al. (2013) destacam que o google speech API fornece suporte para o HTML 5. O recurso é uma nuvem baseado em um serviço no qual os usuários enviam dados de áudio usando uma solicitação de POST HTML e recebem como resposta uma saída em texto do áudio traduzido.
O usuário pode personalizar o número de hipóteses retornada pela ASR, especificar a língua desejada e filtrar determinadas palavras para não serem traduzidas, tais como, palavrões.

3.5.2. AT&TSpeech API

A American Telephone and Telegraph Corporation (AT&T) é uma empresa americana de telecomunicações que fornece serviços de telecomunicação de voz, vídeo, dados e Internet para empresas, particulares e agências governamentais. Na tentativa de tornar a fala a forma dominante das pessoas controlarem a tecnologia, a AT&T disponibilizou sua tecnologia de reconhecimento de voz para que outras pessoas a utilizassem. A tecnologia básica da fala, disponibilizada pela AT&T, é utilizada em muitos de seus próprios aplicativos, incluindo o tradutor AT&T translator app para telefones Android e iOS.

A AT&T Speech API é um serviço baseado em nuvem que pode ser acessado através de solicitações HTML POST, como o Speech Google API. Além disso, pode ser personalizada com dados específicos de um sistema de reconhecimento. Nos testes que foram propostos por Morbini et al. (2013) não foram observadas quaisquer limitações no comprimento para a entrada de dados de áudio. A AT&T não fornece um modelo de linguagem padrão, como no Google, assim os modelos

específicos da aplicação devem ser construídos ou selecionados a partir de uma lista fornecida pelo serviço, sendo a personalização do modelo acústico um dos seus grandes diferenciais.

3.5.3. Apple Dictation

A Apple Dictation é o recurso de sistema operacional, presente no MacOSX e iOS. Está integrado na entrada de texto do sistema, de modo que um usuário pode substituir seu teclado por um microfone para interagir via voz com qualquer aplicação. Associado com o Siri, recurso de assistência pessoal do iOS, ambos os aplicativos compartilham a tecnologia ASR.

O resultado do reconhecimento é uma cadeia de texto com interpretações de palavras individuais ou frases, sendo o seu uso rápido e fácil. O processo de reconhecimento é apropriado para a interação de um único usuário quando, mas de uma pessoa interage, o sistema está sujeito a erros de traduções.

3.6. CONSIDERAÇÕES REFERENTE AS INFORMAÇÕES RELATADAS

O lançamento de produtos com tecnologias de reconhecimento em nuvem como, por exemplo, o Apple Ditaction ou o Google Speech API, fortaleceram a utilização da voz como uma forma moderna de interação. Idosos, em particular, podem apreciar essa modalidade de interação, trocando a utilização do teclado, mouse e demais modalidades não naturais de interação por comandos de voz, facilitando a inclusão digital em seu cotidiano.

A principal vantagem do reconhecimento em nuvem, de um modo geral, é a carga de áudio do pós-processamento. A extração de dados usando uma máquina dedicada e poderosa, que o servidor pode oferecer, é capaz de fazer computação complexa em um curto período de tempo, o que é essencial para os sistemas de reconhecimento de voz em tempo real. Entretanto, este tipo de tecnologia é limitado, uma vez que não possibilita personalização do reconhecimento.

Em contrapartida, o reconhecimento local, como o executado pelo Julius, possibilita controlar totalmente as condições de reconhecimento em tempo de execução, garantindo o retorno dos dados de forma eficiente. Contudo, a degradação no desempenho é mais acentuada em reconhecimento local do que em nuvem, quando existe incompatibilidade de dados.

Na bibliografia é possível obter dados estatísticos que indicam as dificuldades enfrentadas pelos usuários de um ASR, sendo possível constatar que estas tecnologias são úteis, mas que precisam mitigar os problemas. O verdadeiro desafio é adquirir habilidade para convergir as tecnologias em nuvem e local, solucionando assim, as diversidades para a utilização da voz como meio de interação.

Desta forma, conclui-se que o uso de uma tecnologia híbrida é uma excelente forma de facilitar o reconhecimento automático da fala. Não é possível afirmar, a partir desta análise, que este uso misto seja a solução para todos os problemas relacionados a ASR, mas pode-se afirmar que ela é o veículo que pode proporcionar melhorias significativas no reconhecimento e que este tipo de abordagem minimizará os pontos fracos ao mesmo tempo que potencializa os pontos fortes de cada tecnologia envolvida.

4 A VOZ COMO MEIO DE INTERAÇÃO ENTRE SERES HUMANOS E DISPOSITIVOS COMPUTACIONAIS

Este capítulo apresenta o estado da arte na área de interpretação de comandos de voz, tendo como objetivo identificar e classificar, por meio de um levantamento bibliográfico, os principais trabalhos que relatam o uso da voz como meio de interação entre humanos e dispositivos computacionais. O intuito é entender de que forma é abordado o tema e quais as principais contribuições dessa modalidade de interação. O capítulo foi finalizado com a síntese das principais descobertas, que foram utilizadas como fundamentação científica para a elaboração do modelo híbrido proposto nesta dissertação.

4.1. INTRODUÇÃO

O estado da arte demonstrou que o reconhecimento de voz é um tema tratado desde a década de 1980, fato este que pode ser observado através do trabalho de Walsh e Taylor (1987), o mais antigo trabalho científico da área encontrado em todo o processo de análise. Os autores criam um sistema para realizar reconhecimento e análise de voz baseado em microcomputador de baixo custo para extrair as características acústicas da fala. Dois métodos de reconhecimento foram testados, com cada método resultando em pontuações médias de reconhecimento de 95% com um vocabulário de dez palavras.

Muitos estudos têm sido realizados, desde então, em diferentes países, usando a voz como meio para automação de processos, em muitos casos voltados para automação residencial, capazes de ajudar as pessoas em seu cotidiano. Normalmente, os sistemas idealizados dão suporte a vigilância na área de saúde, segurança e conforto, mas também estão presentes em outras áreas como, por exemplo, comunicação entre seres humano e dispositivos computacionais, educação, automobilismo e telecomunicação.

Na última década, observou-se um progresso significativo nas tecnologias de reconhecimento de fala, com a implantação de sistemas cada vez mais independentes do locutor, adaptados para a interpretação da fala contínua e

abrangendo um vocabulário mais extenso. Uma evidência da importância desse tema foi encontrada em Kumar et al. (2011), que descreve um modelo de reconhecimento de voz em nuvem e avalia as suas vantagens e desvantagens ao comparar com o reconhecimento de voz local.

4.2. PLANEJAMENTO E RESULTADO DO ESTUDO

Um levantamento bibliográfico tem como objetivo fornecer uma visão geral de uma determinada área de pesquisa e identificar o tipo e a quantidade de trabalhos disponíveis. Além disso, é imprescindível para a descoberta de padrões, quantidades e frequências de elementos referentes às publicações científicas ao longo do tempo. Nesse sentido, este capítulo visa obter um espectro acerca das publicações do uso de reconhecimento da voz em Amls, bem como, entender como estes trabalhos e as suas contribuições se posicionam no estado da arte do tema.

O objetivo deste estudo é identificar os mais recentes trabalhos que relatam o uso do reconhecimento automático da fala, com o propósito de identificar quais são os mais relevantes e quais tecnologias estão sendo empregadas atualmente.

Com base no objetivo, foram desenvolvidas sete questões de pesquisa que norteiam as investigações deste estudo:

1. Como estão distribuídos cronologicamente os trabalhos a respeito de interpretação de voz em ambientes inteligentes?
2. Qual a distribuição percentual de utilização das plataformas Locais ou em Nuvem?
3. Quais tecnologias estão sendo adotadas para a interpretação de comandos de voz?
4. Quais tecnologias podem interpretar comandos de voz em português brasileiro?
5. Quais as áreas que utilizam interpretações de comandos de voz?
6. Qual é o percentual de utilização de atuadores nos cenários de interação?
7. Quais os tipos de problemas mais frequentes em ambientes com interações via voz?

Com o intuito de não ignorar trabalhos científicos possivelmente importantes para o mapeamento, foram adotadas seis bases científicas:

- ACM Digital Library – em modo “advanced search.”
- IEEE Xplore Digital Library – em modo “command search” e com a opção de pesquisa “Full Text &Metadata. ”
- ScienceDirect – em modo “*expert search*.”
- Springer Link – com os filtros: *content type="article"*, *discipline="computer science."*

Tomando como base as questões de pesquisa foram definidas as principais palavras chaves e seus respectivos sinônimos:

- “voice recognition” – " voice translation” – " voice recognizer” – " voice recognizers” – " voice to text” – " speech recognition” – " speech translation” – " speech recognizer” – " speech recognizers” – "speech to text” – " speech to speech” – "speech-to-speech"
- "speech to text” – " speech to speech”– “voice recognition”

Foram utilizados os operadores lógicos AND e OR, sendo o AND – de caráter exclusivo – utilizado para separar as palavras chaves e o OR – de caráter inclusivo – utilizado para separar os sinônimos de cada palavra chave. Um exemplo da string de busca resultante pode ser observado a seguir:

- (“voice recognition" OR " voice translation" OR " voice recognizer" OR " voice recognizers" OR " voice to text" OR " speech recognition" OR " speech translation" OR " speech recognizer" OR " speech recognizers" OR "speech to text" OR " speech to speech" OR "speech-to-speech") AND ("speech to text” – " speech to speech” – “voice recognition”).

As *strings* específicas foram submetidas às bases de dados e os artigos retornados foram catalogados na ferramenta Mendely, que serviu de apoio para a realização do processo, no período de março a setembro de 2015.

Após o levantamento bibliográfico foram selecionados vinte e um artigos para uma análise mais aprofundada, sendo estes considerados os trabalhos correlatos desta pesquisa. Os autores e os títulos dos trabalhos podem ser contemplados na Tabela 4.

Tabela 4 - Trabalhos selecionados para análise

AUTOR	TÍTULO
Gárate et al. (2005)	Ambient Intelligence as paradigm of a full Automation Process at Home in a real application.
Yi et al. (2007)	Microcontroller Based Voice-Activated Powered Wheelchair Control.
Gao et al. (2007)	Assist Disabled to Control Electronic Devices and Access Computer Functions by Voice Commands.
Mardiana et al. (2009)	Homes appliances controlled using speech recognition in wireless network environment.
López and Callejas (2010)	Multimodal Dialogue for Ambient Intelligence and Smart Environments.
Weiss et al. (2010)	Quality of talking heads in different interaction and media contexts.
Silva et al. (2010)	An open-source speech recognizer for Brazilian Portuguese with a windows programming interface.
Vacher et al. (2011)	The SWEET-HOME Project: Audio Technology in Smart Homes to improve Well-being and Reliance.
Kumar et al. (2011)	Rethinking Speech Recognition on Mobile Devices.
Neto et al. (2011)	Free tools and resources for Brazilian Portuguese speech recognition.
Yu Y. (2012)	Research on speech recognition technology and its application.
Oliveira et al. (2012)	Brazilian Portuguese speech-driven answering system.
Pereira et al. (2012)	A multimedia information system to support the discourse analysis of vídeo recordings of television programs.
Morbini et al. (2013)	Which ASR should I choose for my dialogue system?
Schlogl et al. (2013)	Exploring Voice User Interfaces for Seniors.
Huo et al. (2013)	A Dual-Mode Human Computer Interface Combining Speech and Tongue Motion for People with Severe Disabilities.
Cavallo et al. (2013)	On the design, development and experimentation of the ASTRO assistive robot integrated in smart environments.
AlShu'eili et al. (2014)	Voice Recognition Based Wireless Home Automation System.
Portet et al. (2014)	Design and evaluation of a smart home voice interface for the elderly – Acceptability and objection aspects.
Vacher et al. (2014)	On Distant Speech Recognition for Home Automation.

AUTOR	TÍTULO
Vacher et al. (2014)	The Sweet-Home speech and multimodal corpus for home automation interaction.

Fonte: Autor deste trabalho (2016).

Gárate et al. (2005) desenvolveram um projeto chamado GENIO, o qual através de uma interface multimodal controla qualquer tipo de dispositivo interno, principalmente, os eletrodomésticos e os sistemas de entretenimento. Os autores alertam que o tipo de relacionamento homem-produto, encontrado atualmente baseado em botões, chaves e menus, desaparecerá para ser substituído por sistemas inteligentes, homem-máquina-produto, com os quais vamos interagir através da fala natural, movimentos, gestos, etc. Como resultado do experimento, a interação por voz foi mais aceita que do que a interação por gestos e táteis entre os participantes.

Yi et al. (2007) propuseram um sistema para automatizar cadeira de rodas através da fala. Uma das justificativas apresentadas foi o fato da crescente demanda por cadeira de rodas movidas a eletricidade e de estarem relacionadas com o envelhecimento da população. Como resultado os participantes aprovaram a ideia de estarem com as mãos livres, enquanto se locomovem. Segundo os autores, a tecnologia assistida (AT) é qualquer dispositivo desenvolvido para ajudar as pessoas com deficiência, a fim de executar tarefas que poderiam ser difíceis de serem concluídas.

Gao et al. (2007) ratificam este fato, afirmando que é preciso ajudar as pessoas com determinadas incapacidades físicas, e apresentam um sistema de apoio baseado no reconhecimento de fala. Através do sistema proposto, as pessoas com deficiência podem controlar dispositivos eletrônicos como TV, ventilador, etc., por meio de comandos de voz. O sistema proposto também inclui um emulador de mouse e teclado para ajudar no acesso ao computador e às funções, como navegar na Internet, enviar e-mails e editar um documento.

Outro estudo relevante foi proposto por Silva et al. (2010) que demonstra a utilização de um sistema de reconhecimento da fala para a língua Portuguesa Brasileira. Os resultados são satisfatórios e os recursos para as fases de treinamento e teste deste sistema, tais como, modelos de linguagem e modelos acústicos, estão disponíveis

ao público e podem ser usados em qualquer projeto, desta forma, possibilitando que outras aplicações utilizem os esmos recursos e possam contemplar a linguagem brasileira.

Neto et al. (2011) afirmam que um sistema de reconhecimento automático de fala tem módulos que dependem da língua e, enquanto há muitos recursos públicos para alguns idiomas com, por exemplo, o inglês e o japonês, para outras línguas os recursos são escassos, como é o aso do Português Brasileiro. Nesse sentido, os autores desenvolveram uma aplicação que usa a síntese e reconhecimento de fala em conjunto com um módulo de processamento de linguagem natural que possibilita traduzir conversas do português para o inglês e vice-versa.

Outros dados que chamaram atenção na pesquisa com recursos para o Português Brasileiro foram propostos por Oliveira et al. (2012). Os autores propõem um sistema de atendimento automatizado em *call centers*, com o reconhecimento de fala para o Português Brasileiro. O objetivo do projeto é ser uma alternativa ao uso de teclados numéricos, os quais, segundo os autores, aumentam as taxas de rejeições dos sistemas de *call centers*, de acordo com as análises dos resultados do experimento.

Outro trabalho que contempla a língua Portuguesa brasileira foi proposto por Pereira et al. (2012). O objetivo do projeto foi desenvolver um sistema multimídia para apoiar a tradução de locuções televisivas de outras línguas para o português em tempo real.

Cavallo et al. (2013) desenvolveram um robô com o objetivo de ser integrado a AmIs de forma a auxiliarem idosos em sua vida diária, através de interfaces de voz e táteis. Os resultados do estudo revelaram que os idosos aprovaram a modalidade de interações via voz em detrimento a outras modalidades e escolheram esta forma de interação como sendo a mais adequada para a utilização do protótipo.

AlShu’eili et al. (2014) descrevem a construção de um sistema de automação doméstica via interação por voz. Os testes envolveram 35 indivíduos, de ambos os sexos e com diferentes sotaques. A desenho do estudo determinou que fossem pronunciados 35 comandos de voz no cenário de interação, por participante,

totalizando 1225 comandos. Os resultados demonstraram que o sistema teve 79,8% de comados reconhecidos corretamente, sendo considerada pelos autores uma boa taxa de assertividade.

O projeto de Portet et al. (2014), para automação residencial, foi avaliado por 18 pessoas, com idades entre 60 e 80 anos. Os resultados do experimento mostraram que a maior parte das necessidades dos idosos estão ligadas à confiança e ao aprimoramento da segurança em suas residências. Os participantes demonstraram mais interesse em alertas, relacionados a avisos com relação a emissão de avisos em situações perigosas ou a possibilitar de contactar seus cuidadores em casos de acidentes domésticos e quedas.

Por fim, Vacher et al. (2014) apresentam dois estudos que visam contribuir com o desenvolvimento de um sistema de automação residencial baseado em comando de voz para melhorar o suporte e bem-estar de pessoas com perda de autonomia. O experimento descrito, envolveu 27 participantes, sendo que 11 pessoas deste grupo eram idosas ou pessoas com deficiência visual. Como resultado, um dos maiores problemas identificados foi o tempo de resposta entre a pronuncia de um comando e a sua execução por parte dos dispositivos eletrônicos. A insatisfação dos participantes ficou em torno de 35%, sendo que 42% do público investigado reclamou das constantes repetições dos comandos devido à incapacidade do sistema interpretar as ações desejadas. As limitações técnicas foram reduzidas após o responsável pelo projeto melhorar o gerenciamento de memória do sistema de reconhecimento de voz. Os autores concluíram que apesar do usuário, em alguns casos, necessitar repetir os comandos proferidos, eles estavam, em geral, animados por estarem interagindo com os dispositivos eletrônicos de suas casas via comandos de voz.

A relação de trabalhos apresentados nesta sessão, evidenciam o interesse da comunidade cientifica em realizar novas investigações na área de interpretações de comandos de voz. Os resultados observados confirmam a necessidade de novos estudos para o aprimoramento dos métodos e cenários, sejam estes, fundamentados em ambientes inteligentes ou tomando como base os preceitos oriundos da computação ubíqua e pervasiva.

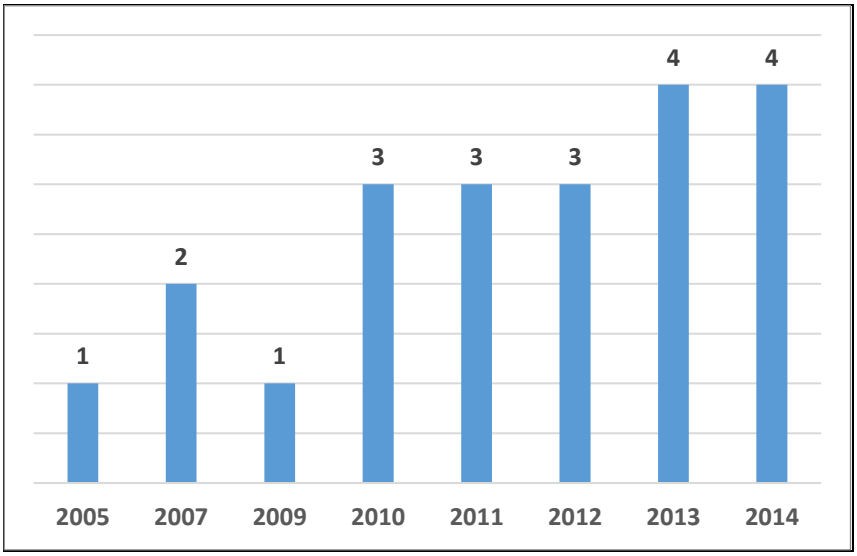
4.3. SUMARIZAÇÃO DO ESTUDO

Esta seção tem como objetivo apresentar alguns dados quantitativos obtidos com a análise dos trabalhos listados na Tabela 4, indicando as tendências e o estado da arte das pesquisas na área de interpretação de comandos de voz. Nas próximas subseções serão respondidas sete questões quantitativas.

4.3.1. Análise referente a distribuição dos trabalhos

A primeira questão tem como objetivo identificar a progressão dos estudos ao longo dos últimos dez anos. A Figura 17 apresenta os 21 artigos selecionados e distribuídos em seus respetivos anos de publicação.

Figura 17– Quantitativo de artigos publicados por ano



Fonte: Autor deste trabalho (2016).

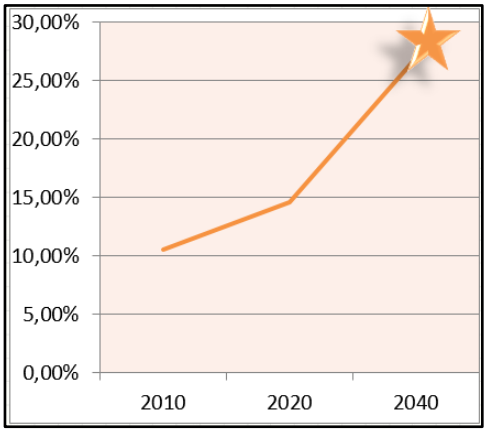
Os dados apresentados na Figura 17 indicam o crescente interesse dos pesquisadores pela área de interpretação de comandos de voz e sua utilização em diferentes cenários. Este interesse pode ser vinculado a evolução de pesquisas em diferentes áreas da computação, entre elas, podem ser destacados, a evolução de ambientes inteligentes, interações naturais e a computação ubíqua e pervasiva.

Além disso, outros fatores sociais são motivadores para a busca de novas formas de interação, em particular, por comandos de voz. Conforme dados do Instituto de

Pesquisa Econômica Aplicada (IPEA) no Brasil (Figura 18) e pela ONG HelpAge (Figura 19), o envelhecimento populacional é um desses fatores que desencadeia pesquisas para que as pessoas possam usufruir de uma boa qualidade de vida após o seu envelhecimento.

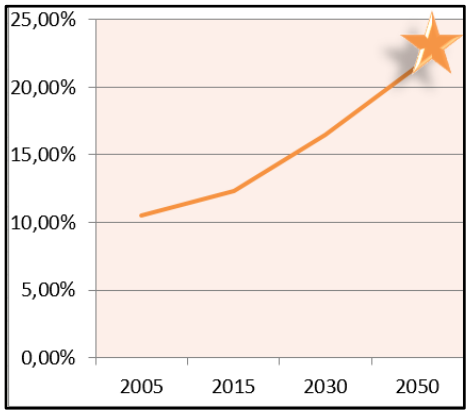
O censo 2010, realizado pelo Instituto Brasileiro de Geografia e Estatística apurou que 45,6 milhões de pessoas declaram possuir algum tipo de deficiência, representando 23,9% da população. A deficiência visual foi a que apresentou maior índice, atingindo 35,8 milhões de pessoas, sendo que a deficiência visual severa, ou seja, aquela em que a pessoa declara ter grande dificuldade de enxergar ou que não consegue enxergar de modo algum, atingia 6,6 milhões de pessoas (IBGE, 2012). Desta forma, comandos de voz é uma alternativa para facilitar a vida diária deste grupo populacional.

Figura 18 - População idosa no Brasil



Fonte: IPEA (30 milhões em 2020).

Figura 19 - População idosa mundial

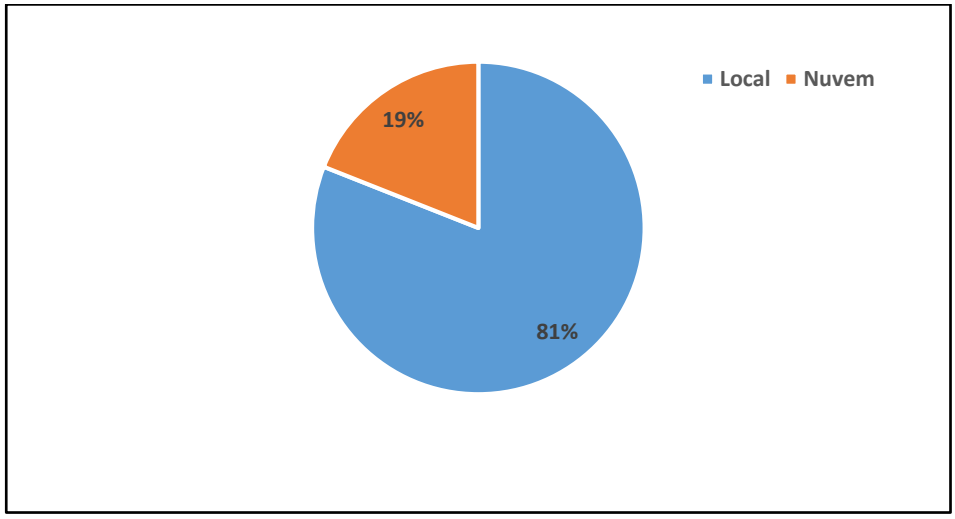


Fonte: HelpAge (1,4 bilhão em 2030).

4.3.2. Análise comparativa da utilização das plataformas Locais ou em Nuvem

O propósito desta análise é identificar nos trabalhos avaliados a tendência em utilizar plataformas em Nuvem ou Locais para realizar as traduções de comandos de voz. No gráfico da Figura 20 pode ser contemplado a divisão percentual nos trabalhos pesquisados, sendo possível perceber ampla preferência pela utilização de plataformas Locais.

Figura 20 - Quantidade percentual de utilização dos dois tipos de plataforma



Fonte: Autoria própria (2016).

A preferência dos pesquisadores por plataformas Locais pode ser justificada pelo fato das tecnologias em nuvem serem recentes, desta forma, existe uma certa restrição a estas plataformas, já que se encontram em um estágio inicial de maturidade. Outros pontos importantes relacionados com a preferência por interpretações locais referem-se à velocidade de resposta, já que não dependem da disponibilidade e velocidade da internet e da vasta documentação das plataformas existentes (KUMAR et al., 2011).

4.3.3. Análise referente as tecnologias utilizadas

A proposta desta subseção é relacionar as diferentes tecnologias existentes para a tradução de comandos de voz com os trabalhos analisados durante a realização

desta pesquisa. A Tabela 5 apresenta as principais tecnologias disponíveis, uma breve descrição e as associa aos projetos desenvolvidos pelo nome dos atores.

Tabela 5 – Principais tecnologias identificadas acerca de reconhecimento de voz

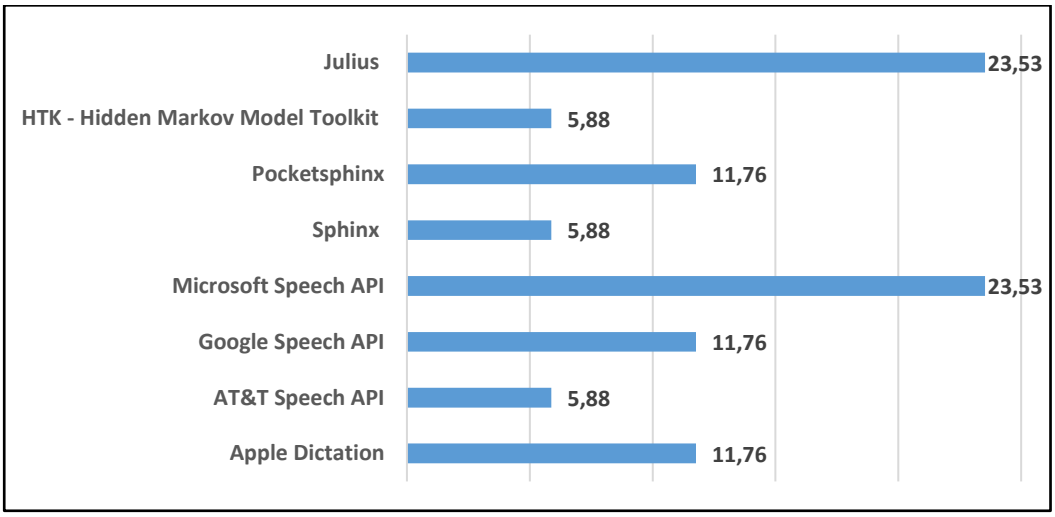
TECNOLOGIA	DESCRIÇÃO	AUTORES QUE ABORDAM
Apple Dictation	Recurso do sistema operacional MacOSX e iOS. Está associado ao Siri, assistente pessoal do iOS.	(MORBINI et al., 2013), (SCHLOGL et al., 2013)
AT&T Speech API	Serviço baseado em nuvem que pode ser acessado através de solicitações HTML POST.	(MORBINI et al., 2013)
Google Speech API	Recurso do Google para reconhecimento de voz em nuvem através de solicitações HTML POST.	(MORBINI et al., 2013), (SCHLOGL et al., 2013)
Microsoft Speech API	Recurso da Microsoft, é possível usar o Speech API Win32 (SAPI) para desenvolver aplicações de voz com o Visual ® Basic, ECMAScript e outras linguagens de automação.	(GAO et al., 2007), (MARDIANA et al., 2009), (ALSHU'EILI et al., 2014), (SILVA et al., 2010)
Sphinx	Ferramenta de código aberto para reconhecimento de voz.	(KUMAR et al., 2011)
Pocketsphinx	Versão otimizada do Sphinx	(MORBINI et al., 2013), (KUMAR et al., 2011)
HTK - Hidden Markov Model Toolkit	Conjunto de ferramentas para a construção e manipulação de modelos ocultos de Markov. Usado principalmente para reconhecimento de voz.	(NETO et al., 2011)
Julius	Ferramenta de código aberto, de alta performance, para reconhecimento de voz.	(SILVA et al., 2010), (OLIVEIRA et al., 2012), (PEREIRA et al., 2012), (NETO et al., 2011)

Fonte: Autor deste trabalho (2016).

No Gráfico da Figura 21 pode ser observado que o Julius e a Microsoft Speech API são as tecnologias mais utilizadas para estudos e projetos na área de interpretação de comandos de voz. Ambas as tecnologias são implementadas em plataformas Locais. Segundo os autores investigados, a principal razão pela preferência por essas tecnologias decorre da flexibilidade de permitir a configuração do vocabulário para interpretar um vasto número de idiomas.

As tecnologias Apple Dictation e Google Speech API são as mais utilizadas para reconhecimento em Nuvem. Este tipo de tecnologia busca formas de melhorar o aprendizado do algoritmo por meio de servidores distribuídos, ou seja, procura formas eficientes de reconhecer vocabulários com um grande número de palavras utilizando processamento distribuído de alto desempenho.

Figura 21 - Distribuição percentual das tecnologias nos trabalhos investigados



Fonte: Autor deste trabalho (2016).

Apesar das tecnologias Sphinx, AT&T Speech e HTK permitirem a criação de novos sistemas de reconhecimento de voz para processamento de comandos e ditados (redação de texto), o fato dessas tecnologias suportarem um número reduzido de idiomas faz com que alguns pesquisadores evitem seu uso como, por exemplo, os brasileiros.

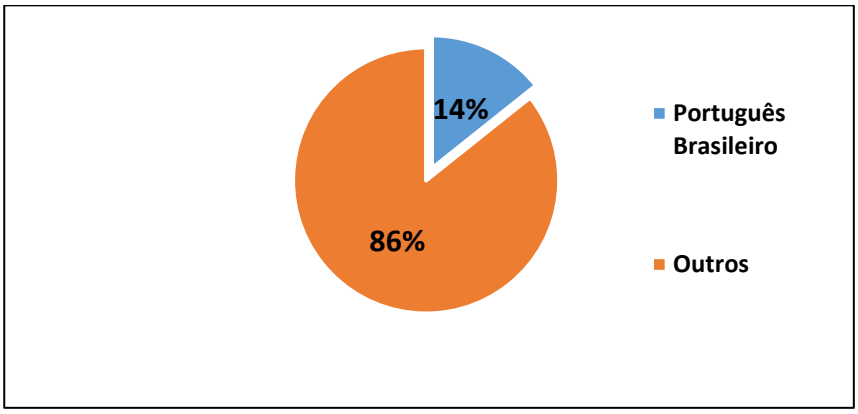
4.3.4. Análise referente a tecnologia para português brasileiro

A Figura 22 indica que o número de estudos relacionados ao reconhecimento automático da fala para o idioma português ainda é reduzido. Um ponto importante a ser observar refere-se a omissão de artigos que não identificam explicitamente o sistema de reconhecimento utilizado ou que optaram por um sistema proprietário/específico para uma determinada função.

Entre as tecnologias investigadas apenas o Google Speech API (plataforma em Nuvem) e o Julius (plataforma Local) disponibilizam recursos para traduzir

comandos de voz oriundos do português brasileiro. Desta forma, essas duas tecnologias foram as escolhidas para a implementação da proposta híbrida que será discutida no próximo capítulo e que representa a maior contribuição desta dissertação.

Figura 22 - Publicações sobre reconhecimento de fala para o português brasileiro



Fonte: Autor deste trabalho (2016).

Para o reconhecimento de voz em português brasileiro no Google Speech API basta informar previamente qual a língua que será utilizada, ou seja, existe uma opção simples de escolha da linguagem. De acordo com a documentação do projeto Coruja, criado pelo grupo FalaBrasil, no Julius é preciso adicionar ao algoritmo os módulos acústicos de linguagem, o que requer trabalho adicional quando comparado ao Google Speech API (SILVA et al., 2010).

4.3.5. Análise referente as áreas de aplicação de comandos de voz

O propósito desta subseção é quantificar a utilização de comandos de voz em diferentes áreas. A Tabela 6 relaciona a utilização de interações via voz com áreas de interesse no contexto desta dissertação. Conforme pode ser observado, a área que mais se beneficia de interações via voz é a Automação Residencial, com 11 trabalhos, entretanto, é possível identificar a utilização da tecnologia em outras áreas afins. Vale ressaltar, que os trabalhos investigados, eram direcionados para a área de Computação Ubíqua e Pervasiva e com o objetivo de acionar componentes eletrônicos ou trazer algum conforto ou qualidade de vida para as pessoas.

Tabela 6 – Áreas de aplicações das tecnologias de reconhecimento de voz

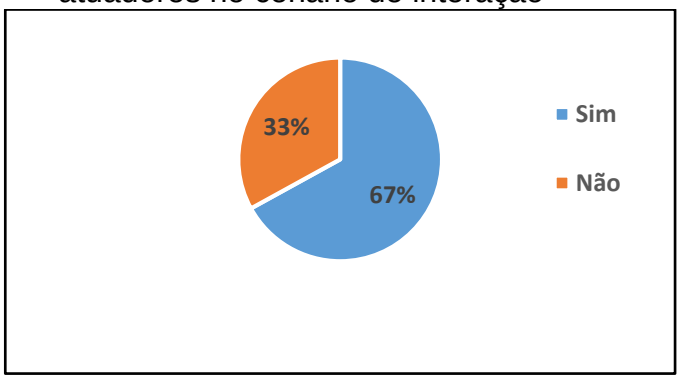
APLICAÇÃO	DESCRIÇÃO	AUTORES QUE ABORDAM
Automação residencial	Automação de casa permitindo controle de equipamentos e eletrodomésticos por voz.	(YU Y., 2012), (MARDIANA et al., 2009), (GÁRATE et al., 2005), (GAO et al., 2007), (LÓPEZ ; CALLEJAS, 2010), (WEISS et al., 2010), (VACHER et al., 2011), (SCHLOGL et al., 2013), (ALSHU'EILi et al., 2014), (PORTET et al., 2014), (VACHER et al., 2014)
Comunicação entre humano e computador	Integração de comandos de voz a aplicações. Integração de reconhecimento de voz a aplicações de dispositivos móveis como celular.	(YU Y., 2012), (WEISS et al., 2010), (KUMAR et al., 2011)
Educação	Auxílio à aprendizagem de idiomas.	(YU Y., 2012)
Saúde	Automação de cadeira de rodas	(YI ET al., 2007)
Telecomunicação	Reconhecimento de voz embutido em ferramenta de mensagem instantânea. Geração automática de legendas em tempo real para programas de televisão.	(OLIVEIRA et al., 2012), (PEREIRA et al., 2012)

Fonte: Autor deste trabalho (2016).

4.3.6. Análise referente a utilização de atuadores no cenário de interação

Na Figura 23, é possível identificar que 67% das soluções utilizam atuadores acoplados a dispositivos eletrônicos para permitir o acionamento dos mesmos. Na automação residencial, o uso de atuadores é vantajoso para o gerenciamento dos dispositivos eletrônicos, uma vez que possibilita que um sistema central (middleware) pode reconhecê-los automaticamente (utilizando uma rede local), mediar a tradução dos comandos de voz proferidos pelo usuário e acionar os atuadores para que executem as ações desejadas pelo usuário. Nesse sentido, o modelo apresentado no próximo capítulo está direcionado para a utilização de atuadores, seguindo a tendência dos pesquisadores investigados.

Figura 23 – Avaliação quantitativa percentual da utilização de atuadores no cenário de interação



Fonte: Autor deste trabalho (2016).

4.3.7. Análise referente aos problemas Identificados

A última investigação referente ao tema de pesquisa tem como objetivo identificar os principais problemas relatados pelos autores dos trabalhos analisados. Na Tabela 7, são relacionados os problemas com os respectivos autores que os abordaram.

Tabela 7– Principais problemáticas identificadas acerca de reconhecimento de voz

PROBLEMA IDENTIFICADO	AUTOR
Acústica dos ambientes (ruído)	Morbini et al.(2013), Kumar et al.(2011), Portet et al.(2014), Vacher et al.(2014), Vacher et al.(2014), Gárate et al.(2005), Mardiana et al.(2009)
Domínio de vocabulários complexos	Morbini et al.(2013), Kumar et al.(2011), Portet et al.(2014), Vacher et al.(2014), Gao et al.(2007), López ; Callejas (2010), Alshu'Eili et al.(2014)
Desempenho no tempo de resposta	Morbini et al.(2013), Kumar et al.(2011), Portet et al.(2014), Vacher et al.(2014), Gao et al.(2007), López ; Callejas (2010)
Personalização do vocabulário	Morbini et al.(2013), Kumar et al.(2011), Portet et al.(2014), Vacher et al.(2014), Alshu'Eili et al.(2014)

PROBLEMA IDENTIFICADO	AUTOR
Qualidade do reconhecimento	Morbini et al.(2013), Kumar et al.(2011), Portet et al.(2014), Vacher et al.(2014), Gao et al.(2007), López ; callejas (2010), Weiss et al.(2010), Vacher et al.(2011)
Idade dos participantes	Vacher et al.(2014), Gárate et al.(2005), Mardiana et al.(2009)
Gênero dos participantes	Vacher et al.(2014), Gárate et al.(2005), Mardiana et al.(2009), Weiss et al.(2010), Vacher et al.(2011)

Fonte: Autor deste trabalho (2016).

Um dos principais problemáticas relacionados com o reconhecimento de voz, tanto em nuvem como local, foi a interferência de ruídos oriundos do cenário de interação. Para contornar o problema, Vacher et al. (2011) propõem o uso de vários microfones para captar o áudio. Segundo suas avaliações, o ASR atingiu bons desempenhos quando o microfone estava próximo do usuário que profere o comando e verificaram que o desempenho do sistema cai significativamente à medida que o microfone é afastado. Esta deterioração ocorre devido a uma ampla variedade de causas, mas principalmente pela presença de ruído de fundo.

Neste contexto, o problema é potencializado quando o cenário de interação é abrangente, por exemplo, envolvendo vários cômodos em uma casa, já que o normal são os usuários se locomoverem no ambiente. Vacher et al. (2014) executaram um experimento utilizando um número crescente de microfones, espalhados em um cenário, tendo constatado que o ponto ótimo para as interpretações da voz no ambiente avaliado chegava ao seu ápice com 21 microfones.

Outro dado que chama atenção está presente na pesquisa de Weiss et al. (2010), a qual investiga o impacto exercido na qualidade do reconhecimento de um sistema de diálogo para casas inteligentes. Como resultado, os autores relataram que as diferenças de gênero impactam nas análises dos experimentos, considerando que o efeito do gênero nas classificações, caso seja ignorado, podem acarretar dados não satisfatórios para os pesquisadores. Outro dado apontado é que o tamanho do

vocabulário para interpretação e a quantidade de pessoas no ambiente também devem ser considerados, pois, influenciam no desempenho e qualidade final do processo.

Vacher et al. (2014) afirmam que um aspecto importante é a diferença de perfil entre os usuários como, por exemplo, o sexo (masculino ou feminino), idade (criança, adulto ou idoso) e a proficiência na língua (linguagem formal versus linguagem coloquial). Essas particularidades trazem implicações para a acústica e impactam nos resultados da ASR.

Gao et al. (2007) constataram que bases dotadas de um vocabulário amplo de palavras aumenta a capacidade do reconhecimento de comandos. Entretanto, o aumento do número de palavras é proporciona a carga de processamento necessária para a realização da interpretação de comandos.

Neste sentido, vem à tona a discussão referente as abordagens projetadas em nuvem versos as metodologias locais (MORBINI et al., 2013). Em nuvem geralmente a base de vocábulos é mais abrangente e consegue resultados satisfatórios devido ao hardware disponível para a realização do processamento. Entretanto, os atrasos são decorrentes da rede de dados, que pode estar congestionada e não prover a tradução em tempo real. Por outro lado, as abordagens locais não sofrem com atrasos de rede, mas normalmente não dispõem de recursos robustos de hardware, o que as limita a trabalhar com um vocabulário mais restrito.

4.4. DISCUSSÃO REFERENTE AOS DADOS LEVANTADOS

O reconhecimento automático da fala é uma área de grande ascensão nas últimas décadas, mas ainda apresenta limitações quanto ao reconhecimento contínuo da fala, sendo esta uma das grandes barreiras a serem ultrapassadas (WEISS et al., 2010). O aprendizado do algoritmo requer grande quantidade de processamento computacional, ficando claro a importância de se propor novas abordagens para o reconhecimento automático da fala com o intuito de propor melhores formas de

interpretação do áudio, levando em consideração as tecnologias disponíveis e suas limitações (GÁRATE et al. 2005).

Neste contexto, a investigação apontou o ASR como sendo o assunto mais abordado direta ou indiretamente nos trabalhos científicos recentes que tratam tema referente a interações em ambientes inteligentes ou mesmo automação de processos. Foi possível observar que os autores, em sua maioria, usaram a ASR para propor alternativas que potencializem a autonomia das pessoas, em geral, pessoas com deficiência ou idosos, amenizando os problemas existentes.

Retomando as sete questões de pesquisa definidas no início deste capítulo, pode-se concluir que:

(i) Os trabalhos relacionados a utilização de comandos de voz está crescendo ao longo dos últimos anos, o que comprova o interesse da comunidade científica em relação a este tema.

(ii) Análise comparativa da utilização das plataformas Locais supera a utilização de plataformas em Nuvem. Entretanto, foi possível constatar que a tendência dos projetos nos últimos anos é utilizar as traduções em Nuvem por abrangeram um vocabulário maior.

(iii) As tecnologias mais utilizadas atualmente para o reconhecimento de voz são o Julius e a Microsoft Speech API, ambas plataformas locais. Contudo, uma tecnologia que vem ganhando destaque é o Google Speech API, por reconhecer um grande variedade de línguas, ter uma plataforma em Nuvem e disponibilizar alto poder computacional.

(iv) Entre as tecnologias mais utilizadas apenas o Julius (plataforma Local) e o Google Speech API (plataforma em Nuvem) disponibilizam interpretações de comandos em português brasileiro.

(v) Foi observado que o reconhecimento de voz é aplicada em várias áreas, mas existe uma tendência forte em utilizar esta tecnologia em ambientes inteligentes, aplicada a automação residencial.

(vi) Os atuadores possibilitam uma grande flexibilidade para a integração dos componentes que fazem parte de um cenário para o reconhecimento de voz, sendo constatado uma tendência na utilização desses componentes

em ambientes inteligentes.

(vii) Os principais problemas enfrentados para o reconhecimento de voz são: acústica dos ambientes (ruído), domínio de vocabulários complexos, desempenho no tempo de resposta, personalização do vocabulário, qualidade do reconhecimento e diferença de perfil das pessoas.

Diante das questões levantadas e visualização do estado da arte na tecnologia de interpretação de voz, algumas decisões foram tomadas para a concepção da proposta que será apresentada no próximo capítulo, entre elas podem ser destacadas:

(i) Será criada uma proposta utilizando ambas as plataformas de reconhecimento de voz, em nuvem e local, caracterizando a proposta em uma abordagem híbrida.

(ii) A personalização do vocabulário ocorrerá em função das palavras mais utilizadas pelos usuários, sendo os comandos interpretados inicialmente em Nuvem e em uma próxima interpretação da mesma palavra, apenas na plataforma Local.

(iii) As tecnologias utilizadas no projeto são as que possuem capacidade de interpretar comandos de voz em português brasileiro, ou seja, Julius (plataforma Local) e o Google Speech API (plataforma em Nuvem).

(iv) Os cenários de interação será integrado a um *middleware* e os dispositivos eletrônicos serão conectados a atuadores.

No próximo capítulo, são utilizadas as afirmações e fundamentações discutidas aqui para a especificação do modelo híbrido proposto.

5 VOICE HOME: MODELO HÍBRIDO PARA O RECONHECIMENTO DE VOZ

Este capítulo aborda a estruturação da plataforma proposta. Inicialmente, serão apresentados os fatores que levaram a construção da plataforma em camadas, além disso, será apresentado o modelo em diagramas UML. Também é abordado o descritivo da plataforma, contemplando os componentes de hardware e software utilizados para o funcionamento da interação via voz em um ambiente residencial.

5.1. INTRODUÇÃO

Conforme constatado no levantamento bibliográfico, o conceito de Automação Residencial com o uso da voz está se popularizando cada vez mais (Morbini et al., 2013). Isso gera novas oportunidades para desenvolver soluções na área de Automação Residencial.

Desta maneira, após a investigação de diversos trabalhos científicos, foi desenvolvido o modelo de uma abordagem híbrida para interações via voz e validada com a construção de uma plataforma em multicamadas independentes, agregando as contribuições de trabalhos anteriores em uma única solução. Deste modo, é possível controlar dispositivos eletrônicos por comandos de voz independente de fabricantes, marcas ou modelos.

Um dos principais objetivos desta plataforma é facilitar a vida diária das pessoas, em particular, daquelas que apresentam restrições de locomoção e idosos. Para a obtenção de uma estrutura plausível aos objetivos desejados, foi criada uma plataforma dividida em três camadas: (i) camada de interação, (ii) camada de processamento e (iii) camada de execução:

- Camada de Interação – Responsável por capturar comandos de voz proferidos no cenário de interação e passar para a Camada de Processamento para que ocorra a interpretação dos comandos.
- Camada de Processamento – Definida por um software correspondente a um *middleware*, responsável pela identificação e comunicação com os dispositivos eletrônicos disponíveis no cenário de interação. Além disso, o

middleware é o responsável pela tradução dos comandos, utilizando uma infraestrutura local ou invocando uma infraestrutura em nuvem para que os comandos possam ser interpretados.

- Camada de Execução – São representados por microprocessadores interligados aos dispositivos eletrônicos que são responsáveis por executar ações no ambiente. Por exemplo, ligar ou desligar a lâmpada, a televisão ou o ar-condicionado.

O restante deste capítulo está subdividido em duas partes principais: (i) a especificação do modelo como solução para a nova abordagem proposta; e (ii) a idealização de uma plataforma para comprovar a viabilidade do modelo e a eficiência da proposta.

5.2. ESPECIFICAÇÃO DO MODELO

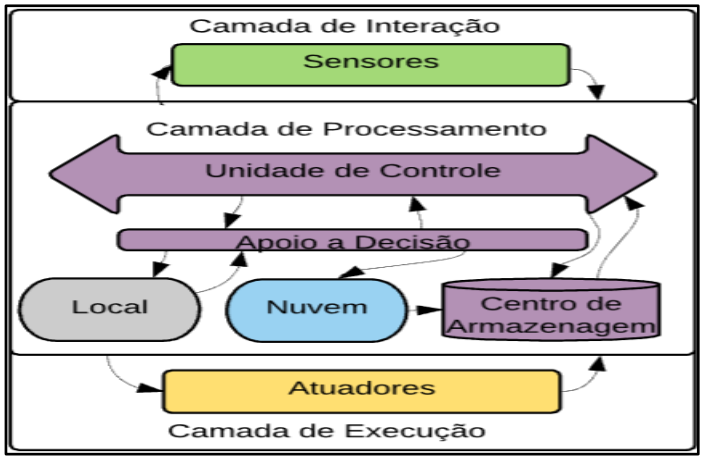
Levando em consideração as abordagens atualmente utilizadas para a tradução de comandos de voz e a necessidade de evoluí-las, os requisitos funcionais do modelo proposto neste trabalho são:

- Permitir a interação do usuário com os dispositivos eletrônicos em um ambiente residencial de interação por intermédio da voz.
- Integrar a voz, atuadores e dispositivos eletrônicos através da Camada de Processamento (*middleware*).
- Identificar e adicionar automaticamente novos atuadores na plataforma.
- Recorrer ao reconhecimento em Nuvem quando o reconhecimento Local não for satisfatório.
- Incrementar a base de dados Local com novos comandos, todas as vezes que for necessário o reconhecimento de comandos com o auxílio da infraestrutura em Nuvem. Desta forma, contribuindo para a personalização do vocabulário Local de acordo com as palavras mais frequentemente utilizadas por seus usuários.

O modelo proposto é dividido em três camadas (Figura 24) para facilitar o desenvolvimento, a separação de interesses e o isolamento de problemas inerentes

aos cenários de interação. A divisão dos componentes em camadas possibilita que a tecnologia empregada possa ser testada com diferentes recursos de hardware e software.

Figura 24 - Imagem do Modelo Proposto

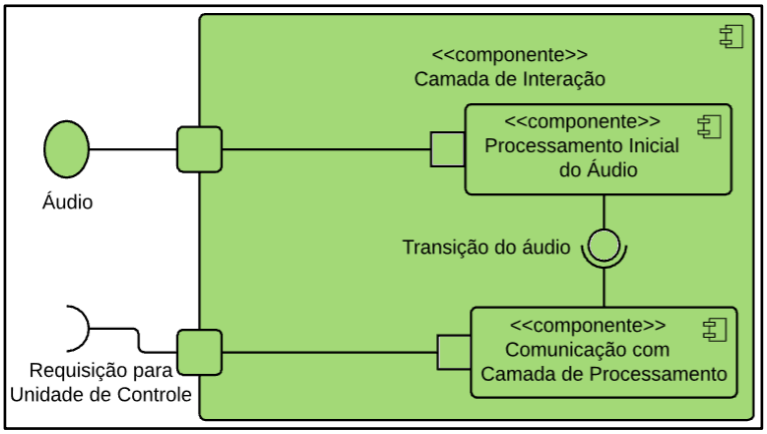


Fonte: Autoria própria (2016).

5.2.1. Camada de Interação

A Camada de Interação tem a finalidade de tratar a captura dos comandos do usuário através da voz para manipular os dispositivos eletrônicos existentes no cenário. Essa camada, está dividida em dois componentes: (i) Processamento inicial do Áudio e (ii) Comunicação com a Camada de Processamento, como pode ser observado na
Figura 25.

Figura 25 - Diagrama de Componentes da Camada de Interação



Fonte: Autoria própria (2016).

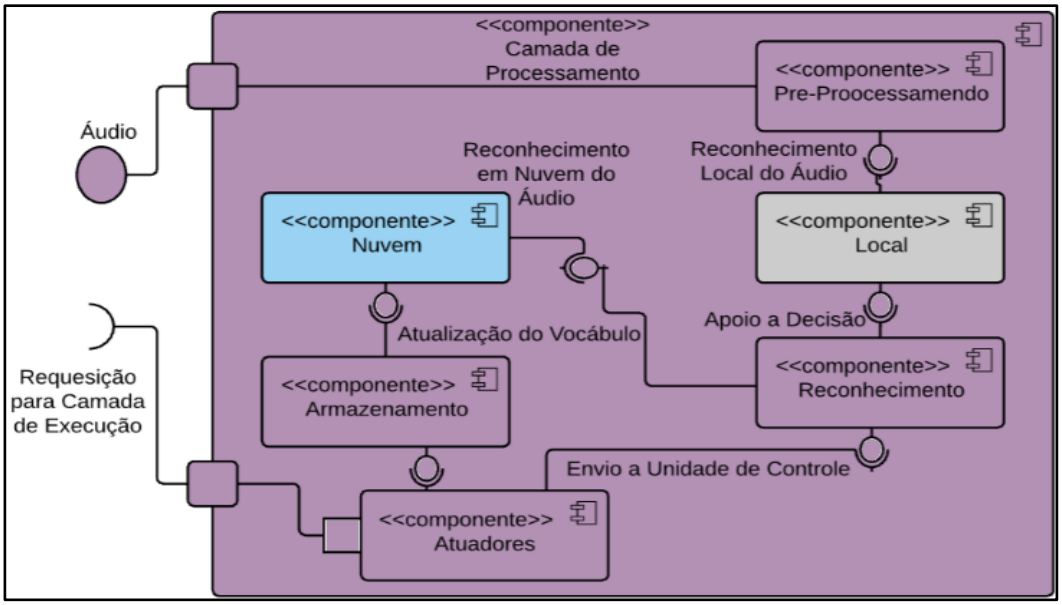
O Componente de Processamento Inicial do Áudio permite a captura automática da voz para reconhecimento. O princípio do seu funcionamento é iniciar o processo de reconhecimento do áudio e disponibilizar para o componente de comunicação com a camada de processamento.

O Componente de Comunicação com a Camada de Processamento é responsável por enviar as requisições de tarefas solicitadas pelos usuários à Camada de Processamento para que o áudio possa ser reconhecido.

5.2.2. Camada de Processamento

A Camada de Processamento é responsável pela detecção dos dispositivos eletrônicos no cenário de interação, por realizar a comunicação entre as camadas e por processar as tarefas requeridas pelos usuários. Essa camada atua como um *middleware* e está dividida em seis componentes conforme apresentado na Figura 26.

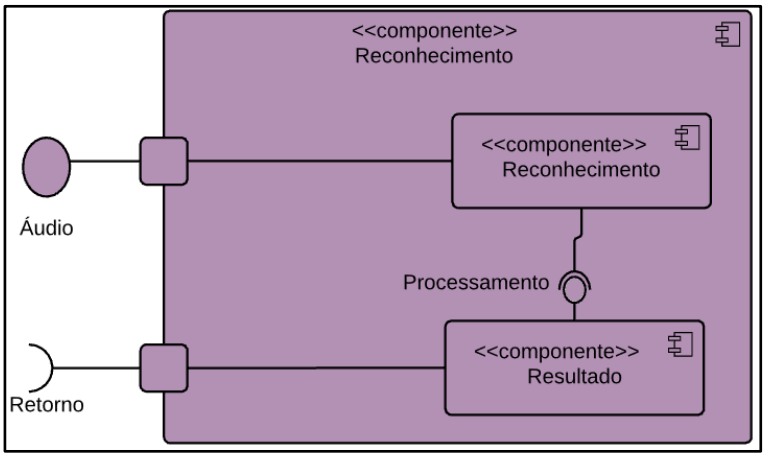
Figura 26 - Diagrama de Componentes da Camada de Processamento



Fonte: Autoria própria (2016).

- Componente Pre-processamento: Responsável por receber o áudio encaminhado pela Camada de Interação e enviar para reconhecimento local.
- Componente Local: O sistema local efetuará o reconhecimento do áudio e enviá-lo para validação no Componente de Reconhecimento.
- Componente de Reconhecimento: O Componente de Reconhecimento é baseado no algoritmo de Markov e retornará três dados: (i) nível de confiabilidade da palavra compreendida; (ii) coeficiente de Viterbi; e (iii) a palavra pronunciada em formato texto para uso da Unidade de Controle. A resposta de retorno do algoritmo será interpretada como bem sucedida se o nível de confiabilidade for acima de 60% e o coeficiente de Viterbi for acima de 50%. Caso o algoritmo não retorne valores indicando o reconhecimento pleno da palavra proferida pelo usuário, será requerida a tradução em Nuvem, caracterizando assim, a proposta híbrida da abordagem defendida nesta dissertação. A Figura 27 demonstra o diagrama.

Figura 27 - Diagrama do Componente Reconhecimento

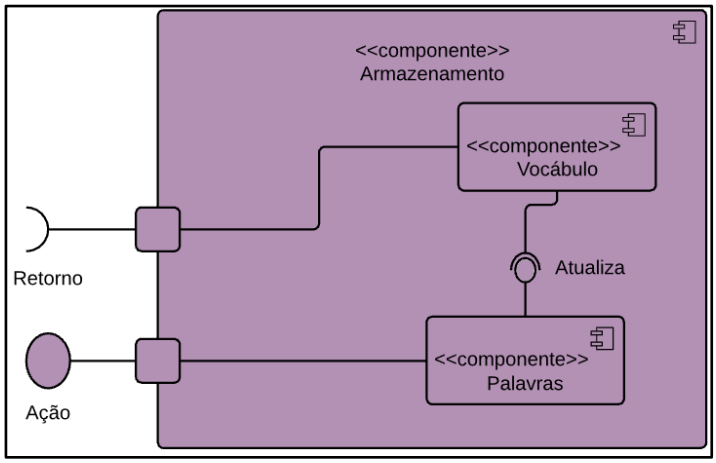


Fonte: Elaborado pelo autor (2016).

- Componente em Nuvem: O sistema em nuvem realiza o tratamento do áudio, quando a reconhecimento Local não for possível. Após o reconhecimento, o componente de Armazenagem é acionado.

- Componente de Armazenamento: Responsável pela distribuição das palavras reconhecidas em nuvem, personalização da gramática e histórico destas ações, conforme demostrado na Figura 28.

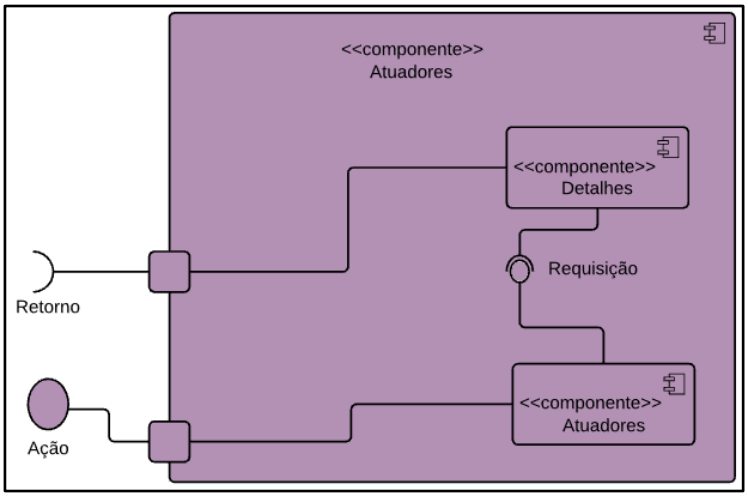
Figura 28 - Diagrama do Componente Armazenamento



Fonte: Elaborado pelo autor (2016).

- Componente Atuadores: Responsável por fazer a listagem automática dos atuadores disponíveis no ambiente, além da identificação das funcionalidades, status que se encontram e acionamento da camada de execução, conforme demostrado na Figura 29.

Figura 29 - Diagrama do Componente Atuadores

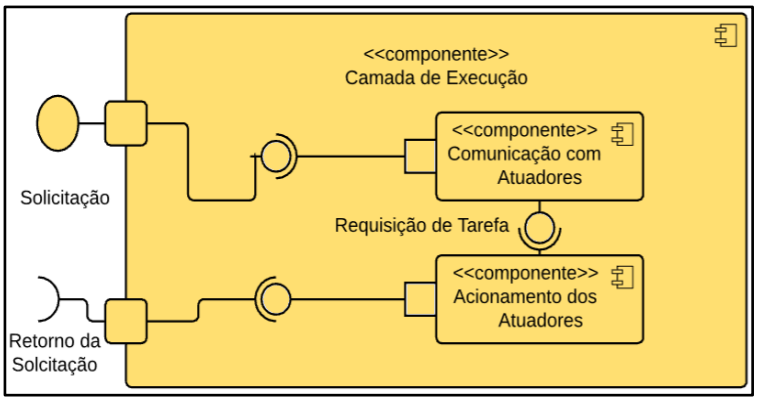


Fonte: Elaborado pelo autor (2016).

5.2.3. Camada de Execução

A Camada de Execução associa os microcontroladores aos dispositivos eletrônicos no ambiente residencial. Isso possibilita que os comandos enviados pelos usuários sejam transformados em ações nos dispositivos eletrônicos. Essa camada está dividida em dois componentes: (i) Comunicação com os Atuadores e (ii) Acionamento dos Atuadores (Figura 30).

Figura 30 - Diagrama de Sequência da Gerência



Fonte: Autoria própria (2016).

O componente Comunicação com Atuadores contém a lógica de acionamento do dispositivo eletrônico, enquanto o Acionamento dos Atuadores é responsável pela comunicação do microcontrolador com esse dispositivo.

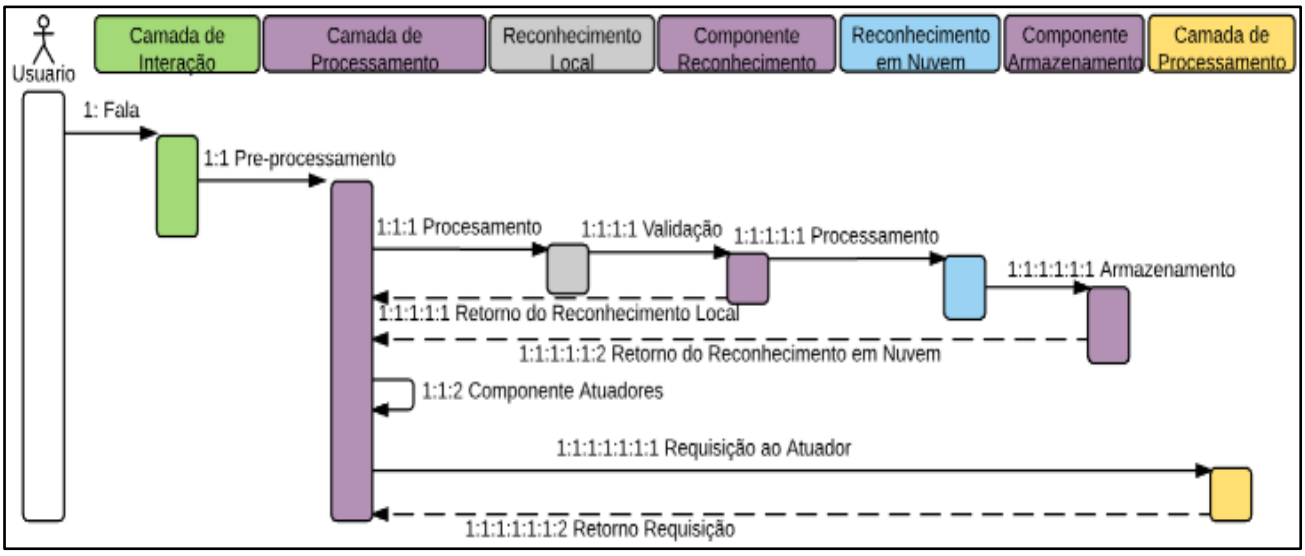
5.2.4. Diagrama de Sequência do modelo proposto

Nesta subseção será demostrado o fluxo de execução do modelo apresentado. A Figura 31 ilustra o diagrama de sequência envolvendo todos os componentes definidos.

O usuário inicia à interação com o sistema, informando o que deseja realizar a partir de um comando de voz. Assim o áudio é captado pela Camada de Interação e enviado para a Camada de Processamento. Nesta camada, o áudio recebido pela Unidade de Controle é enviado para o Reconhecimento Local, onde é feito o processamento da voz. O resultado deste processamento é validado pela unidade

de Apoio a Decisão através do componente de Reconhecimento, de forma a decidir qual a ação mais adequada referente à requisição do usuário.

Figura 31 - Diagrama de sequência



Fonte: Elaborado pelo autor (2016).

Esta análise do fluxo, que define a próxima execução, é baseada no percentual do que foi reconhecido, sendo assim, a saída desde serviço pode resultar em dois tipos de ação: (i) encaminhar o áudio para reconhecimento em Nuvem, sendo esta ação executada sempre que o percentual de reconhecimento não for satisfatório, ou seja, não for possível reconhecer a palavra localmente; e (ii) acionar um dos atuadores existentes no AmI como, por exemplo, ligar uma televisão ou desligar um ar-condicionado, através do Componente Atuadores.

É importante observar que, ao fazer uso do reconhecimento em Nuvem, a palavra que for reconhecida será encaminhada para o componente de Armazenagem, que irá atualizar o vocábulo e deixar o algoritmo de reconhecimento personalizado ao seu utilizador, evitando que no futuro se faça uso da Nuvem para atender a mesma solicitação. Após adaptação da gramática para a reutilização posterior do comando, é acionado um dos atuadores existentes no cenário de interação, através do Componente Atuadores.

Na próxima seção, será apresentada a materialização do modelo em uma plataforma, possibilitando validar a proposta e viabilizar o Voice Home, que permite

a comunicação com diferentes eletrodomésticos e sistemas eletrônicos, independente da marca, tipo ou modelo.

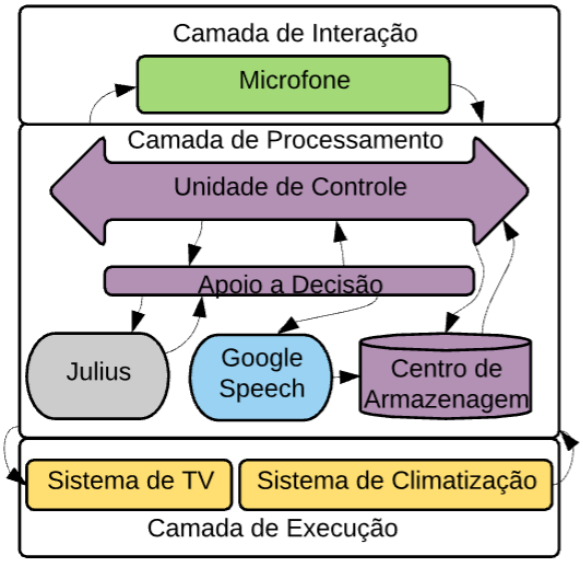
5.3. A PLATAFORMA VOICE HOME

Uma possibilidade para os dispositivos domésticos se integrarem uns aos outros e ficarem visíveis para uma comunicação remota, é conectá-los a uma rede de dados que os identifique por meio de endereços IP. Desta forma, o usuário poderá controlar todos eles através de diferentes modalidades, uma delas é o uso de voz (SATRIA *et al.*, 2015).

Tomando como base a ideia proposta por Rajabzadeh *et al.* (2010) e seguindo as diretrizes definidas no modelo apresentado na seção anterior, neste projeto, foi desenvolvida a plataforma denominada de Voice Home como prova de conceito.

Conforme a proposta do modelo, essa plataforma foi dividida em três camadas (Figura 32): (i) a Camada de Interação, que é representada por um microfone para captura do áudio; (ii) a Camada de Processamento, concebida em um software (*middleware*) e embarcado em um Raspberry Pi; e (iii) a Camada de Execução, que foi idealizada com softwares embarcados em atuadores, utilizando microcontroladores ARM ESP8266, com a função de acionar os dispositivos eletrônicos.

Figura 32 - Estrutura da plataforma dividida em camadas



Fonte: Autoria própria (2016).

5.3.1. Camada de Interação

A Camada de Interação é composta polo módulo de captura do áudio, onde o responsável pela captura da voz é o microfone (Anexo E).

O Módulo de Reconhecimento de Voz é implementado em um RaspberryPi. Para o desenvolvimento deste foi utilizada a biblioteca do Julius que integra o microfone. A biblioteca é projetada para o reconhecimento, em tempo real, de ações que ocorrem em um cenário de interação. O sensor de voz foi implementado através de um microfone diretamente conectado à Camada de Processamento.

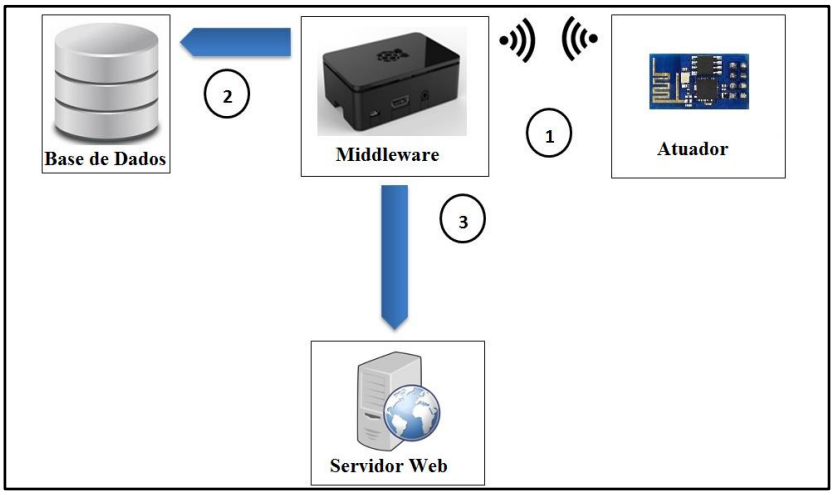
Após a execução de um comando por parte do usuário, o aplicativo inicia uma comunicação através do protocolo HTTP com o *middleware* por meio do seu endereço IP. Então o *middleware* identificará qual foi o comando enviado e acionará o atuador correspondente. Na próxima subseção, serão abordados em detalhes os processos executados pelo *middleware*.

5.3.2. Camada de Processamento

O principal objetivo do *middleware* é realizar a comunicação e disponibilizar serviços entre o Voice Home e os dispositivos eletrônicos (Anexo E). Para prover essa funcionalidade com padronização, foi utilizada a Arquitetura Orientada a Serviços (SOA) (RAMANATHAN; KORTE, 2014).

O *middleware* foi desenvolvido em Java, portanto oferece grande compatibilidade com diferentes plataformas e sistemas operacionais. Nessa prova de conceito, foi utilizada a plataforma Raspberry Pi com o Linux Raspbian. Ele executa três ações (Figura 33): (i) faz uma varredura na rede local para identificar os dispositivos eletrônicos presentes no ambiente; (ii) armazena as informações referentes aos dispositivos; e (iii) inicializa um servidor *Web,* disponibilizando a lista com as informações no formato JSON.

Figura 33 - Módulos do Middleware



Fonte: Autoria própria (2016).

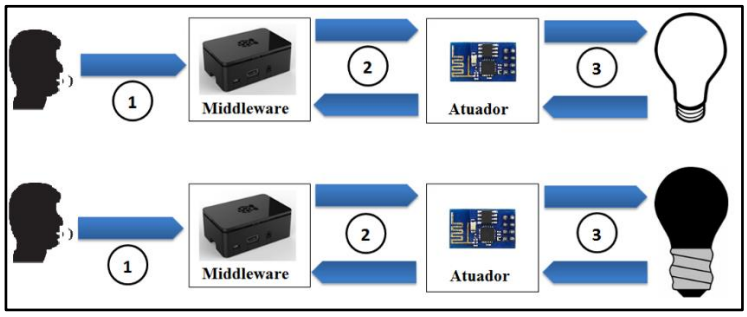
Quando o *middleware* é ligado, ele cria uma conexão com a rede *Wi-Fi* e funciona como um Ponto de Acesso para os atuadores se conectarem. Após essa inicialização, ele realiza uma varredura na rede para encontrar os atuadores e, consequentemente, os dispositivos eletrônicos presentes no ambiente (Passo 1 – Figura 33).

Assim que os atuadores forem encontrados, suas informações serão adicionadas na Base de Dados do *middleware* (Passo 2 – Figura 33). Desta forma, é criada a lista de atuadores no formato JSON, que será disponibilizada por meio de um Servidor *Web* (Passo 3 – Figura 33).

Quando o usuário solicitar uma ação no ambiente por comando de voz, é enviada uma requisição ao *middleware* (Passo 1 – Figura 34), que encaminhará a solicitação para o Atuador correto (Passo 2 – Figura 34), que é um microcontrolador integrado a um dispositivo eletrônico. Tal Atuador realizará a ação e atualizará o status do dispositivo eletrônico (Passo 3 – Figura 34).

A base de conhecimento utilizado na aplicação foi configurada para reconhecer os comandos em português e traduzi-los em ações para interagir com o cenário conforme a necessidade de seu utilizador.

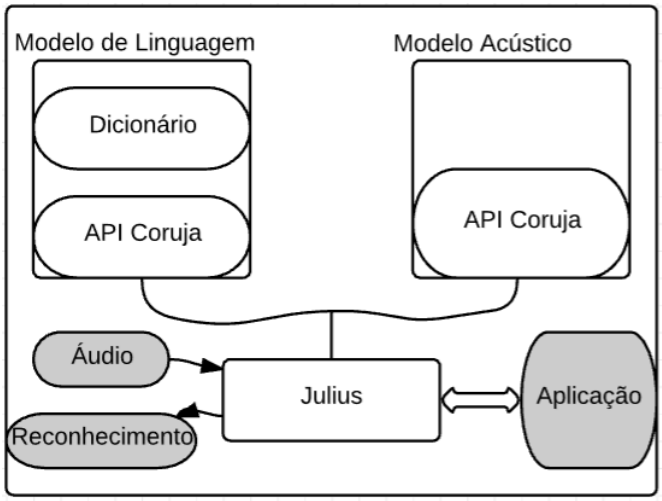
Figura 34 - Exemplo de ações realizadas pela Camada de Processamento



Fonte: Autoria própria (2016).

O algoritmo de reconhecimento Local da fala utiliza o Julius, uma ferramenta independente de idioma, desde que sejam fornecidos o dicionário, modelo de linguagem e modelo acústico. Para que sejam atendidas as necessidades dos brasileiros, o modelo acústico e o modelo de linguagem utilizados são oriundos da API Coruja. O Projeto Coruja é uma API que possibilita o controle em tempo real do decodificador de reconhecimento de voz pelo grupo FalaBrasil do Laboratório de Processamento de Sinais - LaPS[2] da universidade federal do Paraná (UFPA). Na Figura 35, é apresentada a estrutura do Julius.

Figura 35 - Overview Julius



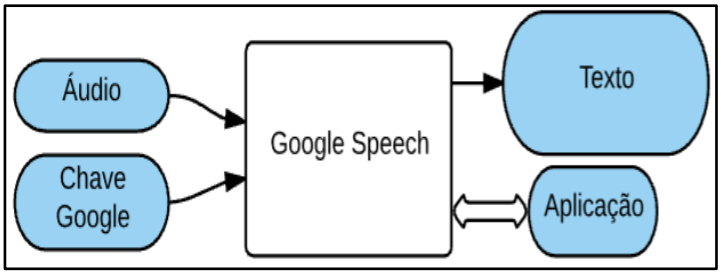
Fonte: Elaborado pelo autor (2016).

O algoritmo de reconhecimento em nuvem utilizado foi o Google Speech[3] v2, a sua escolha foi em função de interpretar a língua portuguesa. Para utilizá-lo é necessário ter uma chave de desenvolvedor do Google. Seu uso é simples e rápido, basta

[2]Disponível em: http://www.laps.ufpa.br/falabrasil/descricao.php
[3] Disponível em: https://cloud.google.com/speech/

enviar o arquivo de áudio agregado a chave do desenvolvedor, via post, para Google Speech. O retorno é uma string indicando o comando traduzido, que será utilizado pela aplicação, conforme demostrado na Figura 36.

Figura 36 - Overview Google Speech



Fonte: Elaborado pelo autor (2016).

Desta forma, o código da camada de processamento irá tratar a entrada do sinal da voz captado pelos microfones com base nos dados do decodificador Julius, intercalará, quando preciso com o segundo algoritmo de reconhecimento em Nuvem, da Google Speech. Quando o reconhecimento em Nuvem for solicitado o retorno da aplicação será usado para atualizar o dicionário do modelo de linguagem do Julius otimizando o reconhecimento local.

5.3.1. Camada de Execução

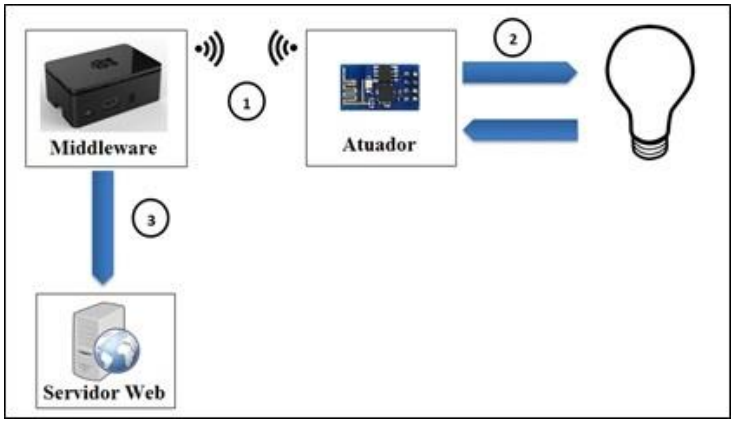
Os atuadores possuem a finalidade de controlar os dispositivos eletrônicos, sendo estruturados em componentes de hardware (microcontroladores) e *firmware* embarcados, que são softwares responsáveis por gerenciar o hardware da plataforma (Anexo E).

As ações desempenhadas pelos Atuadores podem variar entre: (i) funções simples, como ligar/desligar ou abrir/fechar, como no caso de lâmpadas, portas e janelas; e (ii) desempenhar múltiplas funções, como controlar os canais e volume de uma televisão.

Quando um Atuador é ligado pela primeira vez, ele se conectará, via *Wi-Fi,* com o *middleware*, obtendo um endereço IP por DHCP (Passo 1 – Figura 37). Logo após, ele enviará para o *middleware* as informações referentes ao tipo de dispositivo que

irá controlar (Passo 2 – Figura 37). Por fim, o *middleware* atualizará as informações referentes ao dispositivo eletrônico no Servidor *Web* (Passo 3 – Figura 37).

Figura 37 - Funcionamento da Camada de Execução



Fonte: Autoria própria (2016).

Cada atuador terá o seu respectivo endereço IP, ID, *status* e nome. Dessa forma, se ele gerenciar uma televisão, por exemplo, irá se identificar para o *middleware* como tal, utilizando essas propriedades.

Ao receber uma solicitação do *middleware*, o atuador identificará qual deverá ser a ação e irá executá-la, retornando uma confirmação. O *middleware*, por sua vez, irá repassar a informação para o controle remoto universal que identificará se a operação foi realizada com sucesso.

6 RESULTADOS E ANÁLISE DO ESTUDO DE CASO

Neste capítulo, serão apresentados os detalhes da condução do experimento e a análise dos resultados obtidos, possibilitando que possam ser avaliados os pontos fortes e fracos da plataforma Voice Home. O experimento é a oportunidade para a plataforma ser testada em um ambiente real. Toda a sua execução foi realizada tomando como base o *framewor*k DECIDE (SHARP *et al.,* 2007), que viabiliza uma sequência lógica de passos a serem seguidos pelos avaliadores.

6.1. INTRODUÇÃO

O experimento é um importante elemento para validar a ideia ou o produto criado pelo pesquisador. Desta maneira, para a validação da plataforma Voice Home, foi realizado um experimento com trinta pessoas (estudantes e professores) da Universidade Salvador, que responderam a um questionário avaliando a plataforma Voice Home. Com estes dados, foi possível reunir os indicadores necessários para apresentar os resultados finais desta dissertação.

6.2. PLANEJAMENTO DO EXPERIMENTO

O experimento relatado neste trabalho foi dividido em seis fases distintas, tomando como base as diretrizes propostas no *framework* DECIDE (SHARP *et al.*, 2011) que norteou os passos realizados durante todas as fases do experimento:

- **D**eterminar o objetivo da análise – O foco do experimento foi obter informações referentes à usabilidade e a experiência dos usuários com a plataforma. Além disso, sérvio para identificar a percepção dos usuários com relação a eficiência, eficácia do Voice Home. As métricas utilizadas para realizar as avaliações foram adaptadas da proposta original de Kronbauer e Santos (2013), que descreveram métricas para avaliar a plataforma para dispositivos móveis. Os atributos considerados foram: eficiência, eficácia, satisfação, aprendizagem, operabilidade, acessibilidade, utilidade, flexibilidade e facilidade de uso. As descrições dos atributos são apresentadas na Tabela 8.

Tabela 8 - Descrição das Métricas

MÉTRICA	DESCRIÇÃO
Eficiência	Verifica a percepção do usuário com relação ao tempo de resposta da plataforma após a verbalização de um comando.
Eficácia	Mensura a percepção do usuário com relação a assertividade da plataforma para a execução da interpretação dos comandos de voz.
Satisfação	Avalia a satisfação do usuário com relação as interações via voz.
Aprendizagem	Mede a dificuldade que o usuário apresenta para o entendimento do uso da plataforma.
Operabilidade	Avalia a experiência do usuário com a plataforma no sentido de verificar problemas operacionais, tais como, tempo de processamento e atrasos em função da vazão da rede de dados.
Acessibilidade	Mensura a percepção do usuário com relação ao quanto a plataforma pode ser inclusiva para pessoas com algum tipo de deficiência.
Flexibilidade	Mede a versatilidade da plataforma interpretar comandos iguais por meio de palavras diferentes.
Utilidade	Identifica o entendimento do usuário quanto a utilidade da plataforma caso fosse inserida em seu cotidiano.
Facilidade de Uso	Mensura a percepção do usuário com relação a facilidade de utilização da plataforma proposta.

Fonte: Kronbauer e Santos (2013).

- **E**xplorar perguntas a serem respondidas – Tomando como base o objetivo a ser alcançado, foi elaborado um conjunto de perguntas (Tabela 9) que direcionam as análises dos dados, além de comprovar as potencialidades da abordagem proposta neste estudo.
- es**C**olher o método de avaliação – A abordagem escolhida para a obtenção dos dados foi a aplicação de um questionário com a escala de Likert (1932), na qual os valores variam de 1, para muito insatisfeito, até 5, para muito satisfeito.

Tabela 9 - Questões empregadas na avaliação da plataforma Voice Home

PERGUNTAS		ATRIBUTO
1	Qual o seu nível de satisfação com a rapidez (eficiência) com que consegue realizar as tarefas através da voice home?	Eficiência
2	Qual o seu nível de satisfação com a precisão (eficácia) com que consegue executar as tarefas desejadas através da voice home?	Eficácia
3	Qual é seu nível de satisfação com a utilização da voice home?	Satisfação
4	Qual o seu nível de satisfação com a aprendizagem das funcionalidades da voice home?	Aprendizagem
5	Quando houve uma interação indesejada, você conseguiu facilmente retornar para um estado anterior com o objetivo de iniciar a tarefa novamente?	Operabilidade
6	Qual o seu nível de satisfação com a acessibilidade disponibilizada pela voice home?	Acessibilidade
7	Qual o seu nível de satisfação com relação à flexibilidade (caminhos alternativos para executar uma tarefa) disponíveis na voice home?	Flexibilidade
8	Qual o seu nível de satisfação com a utilidade da voice home?	Utilidade
9	Qual a sua percepção com relação a facilidade de uso da voice home?	Facilidade de Uso

Fonte: Autoria própria (2016).

- **I**dentificar e Administrar as questões práticas – Nessa fase, durante a realização do experimento, foram especificados dois documentos utilizados: (i) um texto explicativo referente à proposta do trabalho e dicas simples para a utilização da plataforma Voice Home; e (ii) um roteiro a ser seguido pelo usuário, contemplando as tarefas que devem ser executadas, conforme apresentado no Tabela 10.

Tabela 10 - Roteiro de Ações

ETAPAS	AÇÕES
1	Ligar o ar-condicionado, ajustar a temperatura para um valor confortável e desligá-lo.
2	Ligar a televisão, sintonizar um canal de sua preferência, ajustar o volume e desligar o aparelho.

Fonte: Autoria própria (2016).

O experimento foi realizado no Laboratório de Dispositivos Móveis e Sistemas Embarcados do Pavilhão de Aulas 6 (PA6) da Universidade Salvador (Figura 38 e Fonte: Autoria própria (2016).

Figura **39**). Para construir o cenário de interação, foi necessário utilizar os seguintes recursos: dois dispositivos eletrônicos (televisão e ar-condicionado); dois atuadores esp8266; um microprocessador Raspberry Pi.

Os softwares desenvolvidos para a preparação da plataforma foram embarcados nos seus respectivos dispositivos computacionais e a comunicação entre eles foi disponibilizada via uma rede *Wi-Fi*. Entre os atuadores e os dispositivos eletrônicos, a rede sem fio utilizada foi o infravermelho.

Figura 38 - Experimento da Voice Home – TV Ligada

Fonte: Autoria própria (2016).

Figura 39 - Experimento da Voice Home - TV desligada

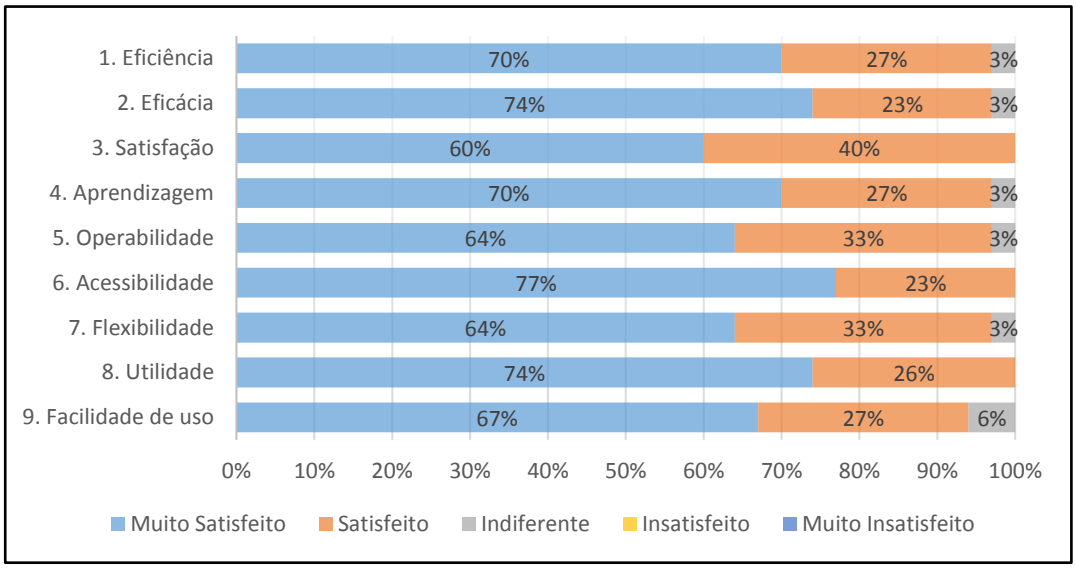
Fonte: Autoria própria (2016).

Para a realização do experimento, foi criado um documento para a leitura do usuário, que é composto de um texto explicando toda a funcionalidade do trabalho e a proposta (Anexo B). Após esse entendimento, foi entregue um roteiro com as tarefas a serem realizadas no cenário de interação (Anexo C).

- **D**ecidir como lidar com as questões éticas – O projeto desta pesquisa foi submetido ao Comitê de Ética, identificado na Plataforma Brasil por meio do identificador CAAE: 51137215.4.0000.5033, sendo devidamente aprovado. O experimento foi conduzido preservando o anonimato dos participantes, pois, no Termo de Consentimento Livre e Esclarecido, está especificado que as informações pessoais dos participantes não seriam divulgadas. Além disso, todos os voluntários possuíam mais de dezoito anos de idade e gozavam de plena capacidade física e mental.

- **E**stabelecer forma de avaliar, interpretar e apresentar os resultados – A coleta de dados ocorreu no período de 04 a 05 de novembro de 2016, com a participação de trinta usuários. O gráfico apresentado na exibe os resultados obtidas após a realização do experimento.

Figura 40 - Apresentação dos Resultados



Fonte: Autoria própria (2016).

6.3. ANÁLISE QUALITATIVA DA PLATAFORMA VOICE HOME

A análise qualitativa tem como principal meta identificar e medir o nível de concordância dos usuários em relação à usabilidade da plataforma Voice Home. A análise foi aplicada através de um questionário contendo nove questões formatadas em função da escala de Likert (1932) e uma questão discursiva, na qual os usuários opinaram, com suas próprias palavras, sobre a plataforma proposta.

As respostas apresentadas pelos usuários geraram um gráfico (Figura 40), onde estão distribuídas as respostas dos usuários através de números inteiros inseridos nas barras. Cada cor utilizada no gráfico representa uma das possíveis respostas referente à escala de Likert (1932) com cinco pontos, representando respostas que variam de muito insatisfeito a muito satisfeito. Estes dados demonstram que o controle remoto universal da plataforma Voice Home agradou a maioria dos usuários que participaram do experimento. Isso comprova que o sistema é eficiente e atende as solicitações dos usuários com um índice muito baixo de falhas.

6.3.1. Avaliação da Eficiência

A Eficiência objetiva avaliar a rapidez com que os comandos são executados pela plataforma após serem proferidos. Nesse quesito, podem ser destacados dois fatores importantes para o sucesso da plataforma: (i) a rapidez de interação do usuário com o cenário de interação, já que os comandos são proferidos via voz; e (ii) a velocidade de processamento do *middleware* e dos atuadores, possibilitando que as ações ocorram em tempo real para contemplar as expectativas dos usuários.

A seguinte pergunta foi direcionada aos participantes do experimento para mensurar a Eficiência:

- Qual o seu nível de satisfação com a rapidez (eficiência) com que consegue realizar as tarefas através da voice home?

O resultado apresentado na Figura 40 refere-se, em valores absolutos, que vinte e um participantes do experimento ficaram muito satisfeitos, oito satisfeitos com o tempo de resposta da plataforma e um indiferente. Com base nesse resultado, pode

ser observado que a proposta foi bem aceita pelos participantes e a resposta dos usuários quanto a Eficiência da plataforma foi positiva.

6.3.2. Avaliação da Eficácia

A Eficácia, neste contexto, mensura a assertividade com que o sistema consegue traduzir os comandos de voz. Quesenbery (2001) acredita que a eficácia avalia como as tarefas foram pontualmente concluídas, e com que frequência elas produziram erros.

Para avaliar este quesito foi proposta a seguinte pergunta:

- Qual o seu nível de satisfação com a precisão (eficácia) com que consegue executar as tarefas desejadas através da voice home?

Os dados obtidos no experimento demonstram que esse atributo obteve vinte e dois participantes muitos satisfeitos, sete satisfeitos e um indiferente. Deste modo, os resultados comprovam que a plataforma tem um alto grau de assertividade na tradução dos comandos de voz.

6.3.3. Avaliação da Satisfação

A satisfação identifica, de modo geral, o sentimento que o usuário tem a respeito de um produto (KRONBAUER ; SANTOS*,* 2013). Assim sendo, a seguinte pergunta foi proposta aos participantes do experimento para avaliar essa métrica:

- Qual é seu nível de satisfação com a utilização da voice home?

De acordo com os dados apresentados na Figura 40, por unanimidade, todos os usuários ficaram satisfeitos na utilização das tarefas, em valores absolutos, dezoito participantes ficaram muito satisfeitos e doze satisfeitos. Pode-se concluir que a Voice Home possui potencialidades que agradam os participantes do experimento.

6.3.4. Avaliação da Aprendizagem

De acordo com Nielsen (1994), o usuário deve assimilar o uso do sistema de forma rápida e fácil. A pergunta proposta aos usuários para avaliar a aprendizagem foi:

- Qual o seu nível de satisfação com a aprendizagem das funcionalidades da voice home?

Os resultados demonstraram que vinte e um voluntários ficaram muito satisfeitos, oito satisfeitos e um indiferente durante o experimento, sendo possível supor que nenhum participante teve algum tipo de dificuldade para executar algum comando desejado.

6.3.5. Avaliação da Operabilidade

Procurando compreender a capacidade da plataforma em se manter funcional, independente das circunstâncias, foi feita uma análise do nível de Operabilidade do cenário de interação. Nesse sentido, foi proposta a seguinte pergunta para os participantes do experimento:

- Quando houve uma interação indesejada, você conseguiu facilmente retornar para um estado anterior com o objetivo de iniciar a tarefa novamente?

Os resultados obtidos mostram que o Voice Home contempla as expectativas da maioria dos usuários. O item revela que dezenove voluntários se mostraram muito satisfeitos com a estabilidade da plataforma, dez ficaram satisfeitos e um indiferente em relação a questões operacionais.

6.3.6. Avaliação da Acessibilidade

Com o intuito de avaliar o atributo de Acessibilidade, o item seis da Figura 40 apresenta o nível de satisfação com relação a este atributo. No caso da Voice Home, os usuários responderam a seguinte pergunta:

- Qual o seu nível de satisfação com a acessibilidade disponibilizada pela voice home?

Os resultados obtidos informam que vinte e três participantes ficaram muito satisfeitos e sete satisfeitos com a acessibilidade da plataforma. Todos os participantes do experimento fizeram uma boa avaliação do sistema em função de acreditarem que a plataforma é adequada para inclusões de pessoas com algum tipo de restrição física.

Com esta informação, pode ser constatado que a plataforma é considerada inclusiva, ou seja, disponibiliza uma nova forma de interação que poderá contemplar idosos e pessoas com necessidades especiais. Como a Acessibilidade é um dos fatores atualmente mais importantes no âmbito da Interação Humano-Computador, com forte apelo social, muitos pesquisadores se dedicam a estudos para a construção de interfaces mais acessíveis (PICCOLO et al., 2011; BRAJNIK, 2006), sendo esta, mais uma contribuição deste trabalho.

6.3.7. Avaliação da Flexibilidade

Para avaliar se a plataforma Voice Home consegue interpretar palavras semelhantes ou com o mesmo significado, foi proposta a seguinte pergunta:

- Qual o seu nível de satisfação com relação à flexibilidade (caminhos alternativos para executar uma tarefa) disponíveis na voice home?

Os resultados indicam que dezenove participantes ficaram muito satisfeitos, dez satisfeitos e um indiferente no que se refere a Flexibilidade do cenário de interação. Neste sentido, pode-se verificar que a Voice Home se comporta de forma adequada, já que o usuário consegue utilizar caminhos alternativos no sistema para realizar uma determinada ação sem apresentar falhas.

6.3.8. Avaliação da Utilidade

Na tentativa de identificar a conformidade entre as tarefas disponibilizadas e os objetivos da plataforma, foi questionada a Utilidade do AmI proposto ao participante caso este fosse inserido em seu cotidiano. Para avaliar esta métrica, foi proposta a seguinte pergunta:

- Qual o seu nível de satisfação com a utilidade da voice home?

De acordo com os dados apresentados no gráfico da Figura 40, verificou-se que todos os participantes acreditam que a plataforma será útil no seu dia-a-dia.

A Utilidade refere-se ao mapeamento das necessidades dos usuários com as funcionalidades do sistema. Este atributo influência diretamente na adoção do sistema (KRONBAUER et al., 2012). Assim, os resultados apresentaram níveis satisfatórios para a utilidade da plataforma proposta.

6.3.9. Avaliação da Facilidade de Uso

Considerando o nível de entendimento de como executar uma tarefa na plataforma, foi realizado o questionamento se o usuário teve alguma dificuldade de realizar alguma das interações propostas. Nesse sentido, a seguinte pergunta foi direcionada aos participantes do experimento:

- Qual a sua percepção com relação a facilidade de uso da voice home?

Percebe-se que a plataforma Voice Home apresenta boa usabilidade, já que dois participantes ficaram indiferentes e vinte e oito não relataram nenhuma dificuldade durante o experimento.

6.4. ANÁLISE DA EFICÁCIA DA PLATAFORMA VOICE HOME

Após a análise qualitativa da plataforma Voice Home em um cenário perfeito, foi necessário realizar a avaliação sobre a métrica da eficácia em diversos cenários para validar o tempo de resposta do sistema. Ou seja, o tempo entre a apresentação de um conjunto de entradas para um sistema (estímulo) e a realização do comportamento desejado (resposta) deve ser satisfatório, caso contrário está sujeito a severas consequências, inclusive de falência da solução apresentada.

O sistema de reconhecimento voz eficaz é aquele cuja performance pode até se degradar, mas não é destruída por falhas em atender as respostas com restrições de tempo. Neste caso, foram propostos dois tipos de análises, conforme apresentado na Tabela 11. Onde será avaliado o tempo de processamento não ocioso, denominado como Fator de Carga Temporal (U), a qual se refere à porcentagem de processamento do CPU durante o processo de reconhecimento. O segundo tipo, demonstrado na Tabela 12, denominado de Fator de Tempo-Real

(xRT), se refere ao tempo que o sistema depende para reconhecer uma sentença. Para complementar esta avaliação foi utilizado quatro tipos de cenários em três períodos distintos (Tabela 13).

Tabela 11 - Fator de Carga Temporal U

Tipificação	Utilização (%)	Zoneamento (Classificação)
A	0 – 25	Poder de processamento excessivo
B	26 – 50	Muito Seguro
C	51 - 68	Seguro
D	69	Limite Teórico
E	70 - 82	Questionável
F	83 - 99	Perigoso
G	100	Sobrecarregado

Fonte: Autoria própria (2016).

A Tabela 11 demonstrará se o sistema ficará sobrecarregado durante sua execução ou mesmo se a CPU é mais poderosa do que o necessário. Em sistemas de reconhecimento em tempo real este fator pode definir o sucesso da aplicação (WILLIAMS, 2006).

Tabela 12 - Fator de Tempo-Real (xRT)

Cálculo	
xRT	Tempo que o sistema depende para reconhecer uma sentença
	Duração da sentença

Fonte: Autoria própria (2016).

Outra métrica para avaliar a eficácia é o fator de tempo-real (xRT), como informado na Tabela 12. O fator xRT e calculado dividindo o tempo que o sistema despende para reconhecer uma sentença pela sua duração. Assim, quanto menor for o fator xRT, mais rápido será o reconhecimento (HUANG et al., 2001).

Tabela 13 – Cenários

Período / Cenários	3G	4G	Rede Doméstica	Rede Doméstica com Netflix
Manhã				
Tarde				
Noite				

Fonte: Elaborado pelo autor (2016).

A definição destes cenários como critérios de avaliação, demonstrados na Tabela 13, visa contemplar os tipos de acesso à internet mais comum no mercado brasileiro (3G, 4G e Rede Doméstica), no seu período de utilização, bem como os habituais processos mais usados, como, por exemplo, é o caso da Netflix, muito usado em ambientes residenciais, em dias atuais.

A proposta é avaliar os tipos de impactos para um eventual reconhecimento de voz nestes cenários identificados e enumerar o que seria viável ou não para sua utilização, abordando o reconhecimento local em separado, bem como apenas nuvem e a proposta desta dissertação que seria local extensível à nuvem.

Quanto ao acesso em nuvem, também será validado a sua utilização com *stream* de vídeo em paralelo, o intuito é prover alguma analise usando '*stress'* comum na rede doméstica e não a sobrecarregar ao ponto de deixá-la inviável para o uso.

Por fim, foi definido que o reconhecimento local seria avaliado com 3 e 65.532 palavras para reconhecimento, o reconhecimento em nuvem e o reconhecimento local com 0 palavra para reconhecimento interligado à nuvem. Esta definição pode ser verificada na Tabela 14.

Tabela 14 – Tipo de reconhecimento

Tipo de reconhecimento	**Situação**
Local com 3 palavras	Cenários
Local com 65.532 palavras	Cenários
Nuvem	Cenários
Local com 0 palavra integrado à Nuvem	Cenários

Fonte: Elaborado pelo autor (2016).

Para obter os resultados sem distorções à sentença, uma vez que para o cálculo do Fator de Tempo-Real (xRT) a duração da sentença é imprescindível, a sentença a ser executada foi gravada e metrificada, conforme apresentação na Tabela 15. Os resultados da avaliação podem ser observados nas próximas subseções.

Tabela 15 - Sentença

Comando a ser executado	Duração
Ligar a TV	1,58 segundos

Fonte: Autoria própria (2016).

6.4.1. Avaliação do Fator de Carga Temporal U

A avaliação do Fator de Carga Temporal U demonstrará se há qualquer restrição na plataforma com o tempo de resposta, nos cenários previamente definidos, de acordo com o processamento utilizado pelo CPU. A proposta é discutir tempos de resposta razoáveis para estes eventos. Desta forma será possível prever se a aplicação tem um poder de processamento adequado para cada cenário.

A captura dos dados foi possível usando o comando "*grep cpu /proc/stat*" em Linux. Conforme definição da Tabela 11, foram avaliados os seguintes cenários dispostos na Tabela 13 e Tabela 14.

O primeiro cenário avaliado foi com um reconhecimento local com um vocábulo reduzido à três palavras para reconhecimento, ilustrado na Tabela 16. De acordo com os dados obtidos é possível notar que o período não causa interferência significativa quanto ao reconhecimento e o uso da CPU ficou com a tipificação A, em números significa que o seu uso foi entre 0% a 25% o que significa que existe um poder de processo muito grande para a demanda dos dados analisados.

Conforme já discutido em seções anteriores quanto maior o vocábulo para reconhecimento melhor é a satisfação do usuário, uma vez que o usuário terá liberdade para diversos tipos de interação, apesar de que a limitação drástica do vocábulo, como neste cenário pode ser útil para alguma aplicação mais simples e

especifica, as quais não precisam de uma maior flexibilidade de comunicação, como, por exemplo, um elevador com apenas um andar, onde é preciso solicitar que o mesmo suba ou desça.

Tabela 16 - Local com 3 palavras

Fator de Carga Temporal U	3G	4G	Rede doméstica	Rede doméstica com Netflix
Manhã	A	A	A	A
Tarde	A	A	A	A
Noite	A	A	A	A

Fonte: Autoria própria (2016).

No segundo cenário avaliado foi mapeado 65.532 palavras do vocabulário português do Brasil para reconhecimento. É notado na Tabela 17 que o processo da CPU ficou na tipificação G, ou seja, o seu processamento ficou no limite, sobrecarregado. Desta forma fica claro que a quantidade de palavras a ser reconhecida influencia no poder de processamento nas análises de reconhecimento dos dados. Outro ponto a ser observado é que o período também não influenciou os testes.

Tabela 17 - Local com 65.532 palavras

Fator de Carga Temporal U	3G	4G	Rede doméstica	Rede doméstica com Netflix
Manhã	G	G	G	G
Tarde	G	G	G	G
Noite	G	G	G	G

Fonte: Autoria própria (2016).

A Tabela 18 demonstra os testes efetuados no terceiro cenário de reconhecimento, a Nuvem. Os dados obtidos apontam a tipificação B com utilização do processamento da CPU entre 26% e 50% classificando a aplicação como muito seguro para utilização de rede doméstica ou rede doméstica com o uso da Netflix. Neste cenário o período é um fator importante, por exemplo, a noite teve um maior uso da CPU em ambos os reconhecimentos, chegando a mudar de tipificação, para C na rede doméstica com o uso da Netflix, este tipo de tipificação é classificado quando o uso da CPU está entre 51% e 68% de sua utilização, ainda assim, de acordo com a classificação é um sistema seguro. Quanto à utilização do

reconhecimento nas redes 3G e 4G o resultado foi deteriorante, permanecendo na classificação questionável, quando a sua utilização fica entre 70% e 82%.

Tabela 18 - Nuvem

Fator de Carga Temporal U	3G	4G	Rede doméstica	Rede doméstica com Netflix
Manhã	E	E	B	B
Tarde	E	E	B	B
Noite	E	E	B	C

Fonte: Autoria própria (2016).

Para o quarto cenário a ser analisado foi proposto o reconhecimento local, sem nenhuma palavra para reconhecimento, sendo assim não seria possível identificar a solicitação mencionada e a nuvem seria usada para reconhecimento. O intuito era forçar o uso da nuvem para que o reconhecimento fosse concluído com sucesso. De acordo com a Tabela 19 os resultados foram praticamente iguais à validação anterior. Houve uma certa divergência de resultados, mas nada que fizesse com que a faixa da tipificação fosse alterada.

Este resultado demonstra que o uso do reconhecimento local alinhado à nuvem não é mais degradante que apenas o uso da nuvem e quando este procedimento obter o reconhecimento via processo local terá seus resultados próximos ao da Tabela 16 demonstrada no início das análises.

Tabela 19 –Local com 0 palavra integrado à Nuvem

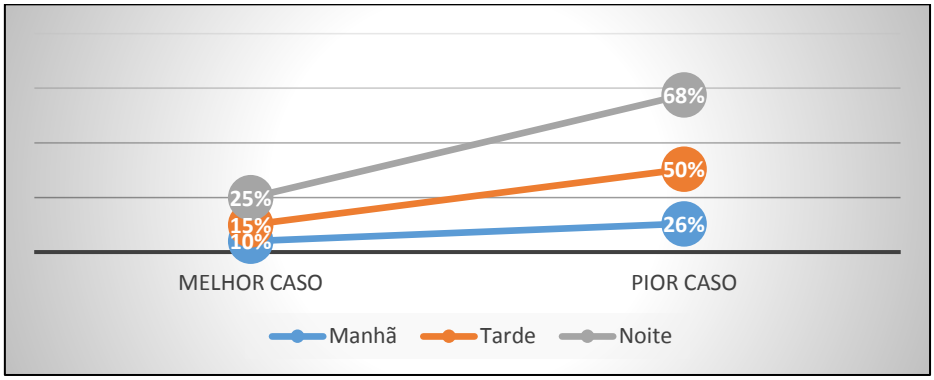
Fator de Carga Temporal U	3G	4G	Rede doméstica	Rede doméstica com Netflix
Manhã	E	E	B	B
Tarde	E	E	B	B
Noite	E	E	B	C

Fonte: Autoria própria (2016).

A Figura 41 demonstra que, de acordo com os resultados analisados, o reconhecimento local alinhado à nuvem pode ser a melhor opção para reconhecimento desde que existe uma rede doméstica disponível para acesso à internet com fluidez, sendo assim, em melhor caso, passa a ser quando o

reconhecimento for via método local, nesta situação o uso da CPU ficará entre 0% a 25%, com a tipificação A e no pior caso quando a palavra não for reconhecida localmente e for preciso usar a nuvem para reconhecimento, sendo assim o uso da CPU ficará entre 26% a 68%, variando entre a tipificação B e C. Sendo a tipificação C mais comumente encontrada quando o reconhecimento em nuvem for em um período da noite. Normalmente neste período existe um maior número de usuários conectados e esta situação tente a piorar quando na mesma rede tem acesso a outros serviços que exijam dados, como é o exemplo de servidores de *stream* de vídeo ou áudio em paralelo, Netflix, Spotify, entre outros.

Figura 41 – Análise dos cenários do Fator de Carga Temporal U



Fonte: Autoria própria (2016).

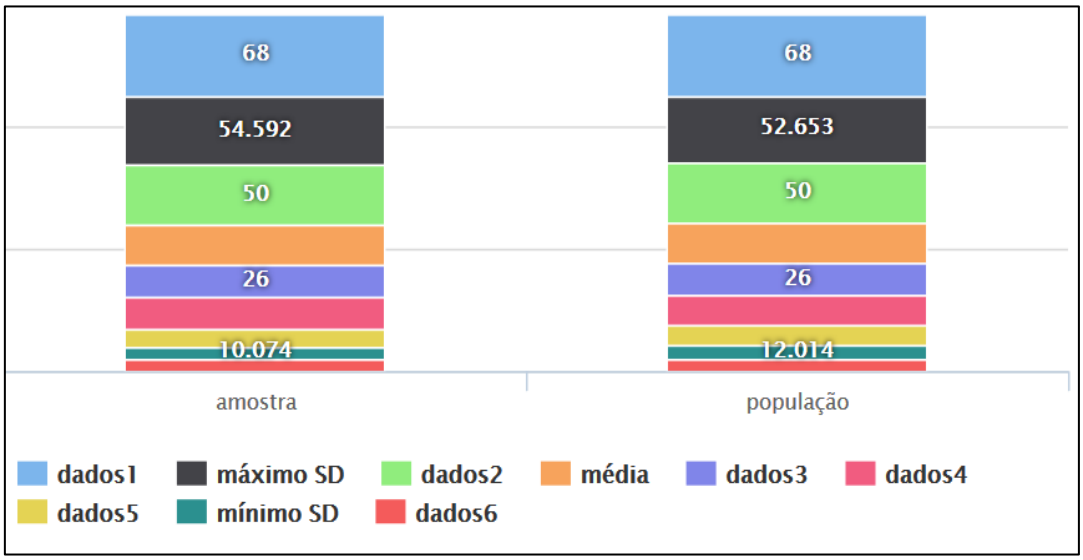
Com base nos resultados apontados na Figura 41, foi possível calcular o Desvio Padrão (DP) dos cenários apresentados. O DP é a medida de dispersão dos números em um conjunto de dados partindo de seu valor médio.

Desvio Padrão: $S=\sqrt{\frac{\sum(X-M)^2}{n-1}}$

População Desvio Padrão: $S=\sqrt{\frac{\sum(X-M)^2}{n}}$

Desta forma a Figura 42 demonstra uma medida da variabilidade ou da volatilidade do conjunto de dados apresentado, obtendo dados como a media 32%, desvio padrão em 22% e a população desvio padrão com 20%. Os dados usados para os cálculos apresentados a seguir, foram os três números do melhor caso e os três números do pior caso aplicados à formula da DP.

Figura 42 - Desvio Padrão do Fator de Carga Temporal U



Fonte: Autoria própria (2016).

6.4.2. Avaliação do Fator de Tempo-Real (xRT)

O segundo tipo de análise proposto tem sua utilização muito comum para mensurar tempo de resposta dos processos de reconhecimento de voz. A formula é obtida dividindo o tempo que o sistema depende para reconhecer uma sentença pela sua duração, conforme demonstrado na Tabela 15, a duração obtida, com a sentença gravada, foi de 1,58 segundos. Assim quanto menor for o xRT, mais rápido será o reconhecimento.

A forma de captura dos dados foi obtida através do comando *"System.nanoTime()",* de forma que foi obtido o tempo antes e depois do reconhecimento, subtraindo o tempo final pelo tempo inicial para chegar ao resultado esperado.

Em sua primeira avaliação, a qual foi utilizada o cenário proposto localmente com 3 palavras para reconhecimento, houve uma variação média de 2,88 a 2,96 segundos, conforme demonstrado na Tabela 20. Com este resultado é possível afirmar que o período da noite se tem um reconhecimento mais rápido do que o período da manhã para o reconhecimento local.

Tabela 20 - Local com 3 palavras

Fator de Tempo-Real (xRT)	3G	4G	Rede doméstica	Rede doméstica com Netflix
Manhã	2,962	2,962	2,962	2,962
Tarde	2,917	2,917	2,917	2,917
Noite	2,886	2,886	2,886	2,886

Fonte: Autoria própria (2016).

O segundo cenário de avaliação proposto foi o local com 65.532 palavras para reconhecimento, demonstrado na Tabela 21. As análises dos resultados sugerem que quanto mais palavras conter para reconhecimento, mesmo que o comando seja o mesmo, o tempo de resposta irá se degradar de acordo com a quantidade de palavras a ser validada. Em alguns resultados o tempo de resposta quase dobrou e novamente o período noturno se teve o reconhecimento mais rápido, quando comparado com os demais períodos desta avaliação.

Tabela 21 - Local com 65.532 palavras

Fator de Tempo-Real (xRT)	3G	4G	Rede doméstica	Rede doméstica com Netflix
Manhã	5,550	5,550	5,550	5,550
Tarde	5,467	5,467	5,467	5,467
Noite	4,945	4,945	4,945	4,945

Fonte: Autoria própria (2016).

A Tabela 22 apresenta o resultado do terceiro cenário avaliado. De acordo com os resultados o período da manhã possui o tempo de resposta mais rápido em todos os tipos de reconhecimento analisados. O tempo nas redes de acesso à internet 3G e 4G tem o tempo mais degradado se comparado com um acesso à rede doméstica, onde praticamente levou o dobro de tempo para o mesmo reconhecimento, chegando à 9,73 no tempo mais longo e 4,53 no tempo mais rápido. É notado também que a rede doméstica com uso em paralelo de outros serviços que demandem grande quantidade de serviços de dados tem o tempo de resposta prejudicado.

Tabela 22 - Nuvem

Fator de Tempo-Real (xRT)	3G	4G	Rede doméstica	Rede doméstica com Netflix
Manhã	9,651	8,323	4,533	4,929
Tarde	9,663	8,349	4,844	5,386
Noite	9,736	8,353	5,189	6,128

Fonte: Autoria própria (2016).

No último cenário avaliado, demonstrado na Tabela 23, podemos afirmar que, novamente, como demonstrado no primeiro item avaliado desta seção 6.4, o reconhecimento local alinhado à nuvem para reconhecimento tem um retorno positivo, sendo possível concluir que em um pior caso, este tipo de reconhecimento irá demorar entre 1 a 2 segundos do que o reconhecimento em nuvem. Chegando em seu melhor caso na mesma situação apresentada na Tabela 20, quando o reconhecimento for resultante com sucesso via método local. É importante destacar que este procedimento só é viável caso se tenha um cenário com acesso à rede doméstica e que mesmo que concorrência em paralelo como é o caso da Netflix os resultados foram satisfatórios.

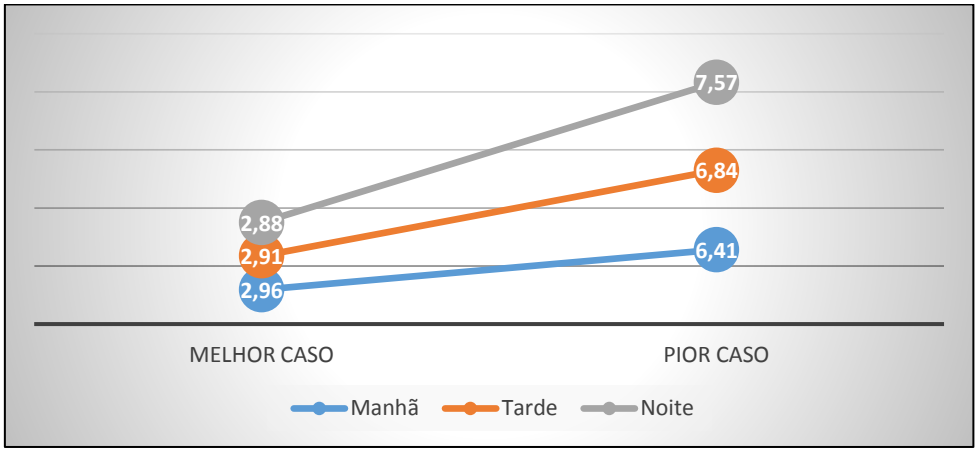
Tabela 23 –Local com 0 palavra integrado à Nuvem

Fator de Tempo-Real (xRT)	3G	4G	Rede doméstica	Rede doméstica com Netflix
Manhã	11,132	9,804	6,014	6,410
Tarde	11,122	9,807	6,303	6,844
Noite	11,179	9,796	6,632	7,571

Fonte: Autoria própria (2016).

Na Figura 43 é demonstrado o melhor e o pior caso para as situações apresentadas. Onde de acordo com os resultados, como apresentado na subseção 6.4.1., o reconhecimento local alinhado à Nuvem tem o melhor resultado dentre os apresentados. Ou seja, o tempo de resposta pode variar entre 2,96 a 2,88 segundos de acordo com período analisado para o melhor caso e obtendo um tempo entre 6,41 a 7,57 segundos no pior caso.

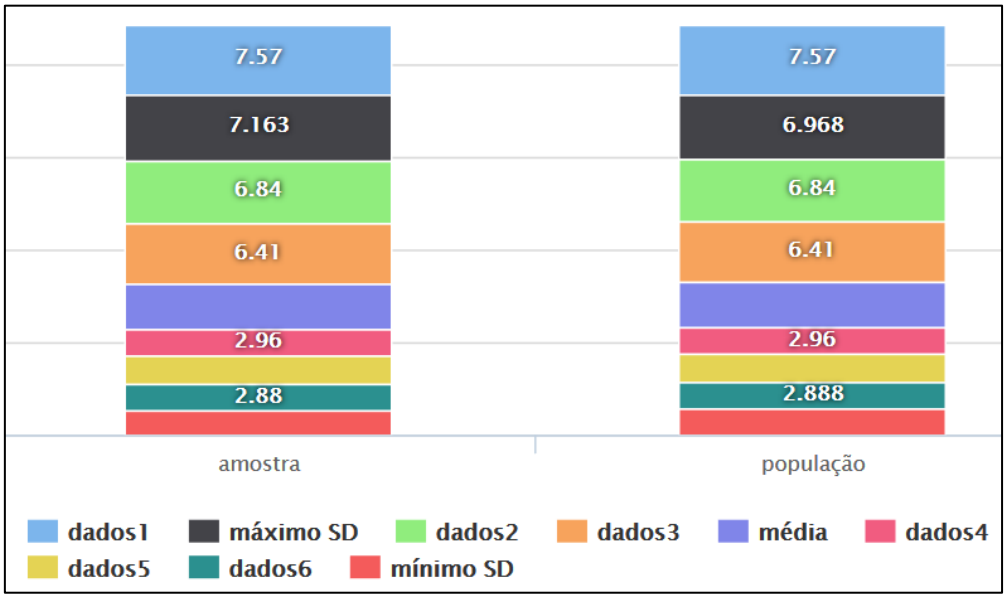
Figura 43 - Analise dos cenários do Fator de Tempo-Real (xRT)



Fonte: Autoria própria (2016).

A avaliação destes cenários foi concluída obtendo o DP deste cenário, como informado na Figura 44 a média foi de 4.928 segundos, o desvio padrão com 2.234 segundos e a população do desvio padrão de 2.040 segundos. Os seis dados usados são os números apresentados no melhor e pior caso e a fórmula usada é a mesma mencionada na subseção anterior.

Figura 44 - Desvio Padrão do Fator de Tempo-Real (xRT)



Fonte: Autoria própria (2016).

7 CONCLUSÕES E TRABALHOS FUTUROS

Neste capitulo são descritas as conclusões da pesquisa realizada apresentando uma discussão sobre a validação, descrevendo as conclusões obtidas e sugerindo recomendações para trabalhos futuros.

7.1. CONCLUSÕES

A intenção desse trabalho foi desenvolver uma solução que forneça mais qualidade de vida para as pessoas, principalmente aquelas com dificuldade de movimento, idosos e crianças. Com a utilização do Voice Home, é possível oferecer uma solução que contempla conforto e comodidade para o usuário de forma que ele tenha o poder de gerenciar os dispositivos eletrônicos através da voz com comodidade e simplicidade.

Este trabalho também apresenta o resultado do experimento que visa à aceitabilidade de Ambientes Inteligentes (AmI) equipado com aplicações tecnológicas de reconhecimento de voz, trazendo uma abordagem tecnológica híbrida, a qual contempla algoritmos de interpretação de comandos de voz Local e em Nuvem, buscando mais eficiência na compreensão da fala. Além disso, o modelo proposto pode ser:

- Adaptado a vários ambientes
    - Possibilitando atuar em cenários diversificados.
- Permite a interação com diversos dispositivos eletrônicos
    - Independentemente de suas marcas ou modelos.
- Compatível com outras formas de interação.
    - A plataforma permite integrar outras modalidades de interação.

Inicialmente, com o intuito de entender as áreas transversais a este trabalho, as tecnologias disponíveis para a interpretação de voz e as aplicações que utilizam esta modalidade de interação, foi realizado um levantamento bibliográfico abrangente que descreve os principais tópicos referente ao assunto.

Após a descrição das áreas transversais, foi realizado um estudo sobre o estado da arte onde foram selecionados e analisados 21 trabalhos que abordam AmI com interações via voz. Esta análise aborda:

- Utilização das plataformas Locais ou em Nuvem.
- Tecnologias adotadas para a interpretação de comandos de voz.
- Tecnologias que interpretam comandos de voz em português brasileiro.
- Áreas que utilizam interpretações de comandos de voz.
- Utilização de atuadores nos cenários de interação.
- Tipos de problemas mais frequentes em ambientes com interações via voz.

Como forma de criar uma solução completa, nesta dissertação, foi proposta uma abordagem que contempla um modelo e sua materialização em uma plataforma para controlar todos os dispositivos em uma residência. A estrutura do modelo que permite essa comunicação é composta de três camadas. Ela está alicerçada no modelo de múltiplas camadas, proposto por Edwards e Foreword (1998). Inspirado nele, foram criadas as seguintes camadas:

- A Camada de Interação prevê as interações do usuário via voz.
- A Camada de Processamento é responsável pelo gerenciamento e controle do fluxo dentro da plataforma.
- A Camada de Execução é responsável em acionar os dispositivos eletrônicos existentes no cenário de interação.

Para validar a proposta, foi realizado um experimento em laboratório envolvendo 30 usuários, tomando como base as diretrizes propostas pelo framework DECIDE. O seu objetivo foi identificar o nível de satisfação do usuário dentro da plataforma, utilizando nove atributos de usabilidade definidos por Kronbauer e Santos (2013):

- Eficiência
- Eficácia
- Satisfação
- Aprendizagem
- Operabilidade
- Acessibilidade
- Flexibilidade

- Utilidade
- Facilidade de Uso

Com os resultados da execução do experimento, é possível concluir que a plataforma pode ser uma boa alternativa para a criação de AmIs com a modalidade de interações via voz. Os resultados observados apresentam, na média geral, uma satisfação acima de 60% em todos os seus atributos.

As métricas relacionadas com a Eficiência, Satisfação e Utilidade da plataforma obtiveram percentuais de aceitação de 99%. O que indica a possibilidade de a proposta ser utilizada em larga escala. Além disso, sugere a viabilidade de ser empregada para auxiliar pessoas com necessidades especiais ou restrições de locomoção.

Para validar à eficácia proposta no tema desta dissertação, foi realizado várias análises quanto à performance do reconhecimento em cenários diversificados, tomando como base as diretrizes propostas pelo Williams (2006) e pelo Huang (2001). O seu objetivo foi identificar em quais cenários o tema proposto é viável:

- Fator de Carga Temporal U
- Fator de Tempo-Real (xRT)

A partir dos resultados, percebeu-se que o tema defendido neste projeto atendeu de maneira satisfatória aos anseios de performance do experimento, já que os resultados obtidos foram positivos nas duas métricas avaliadas para mensurar à eficácia proposta.

Desta forma, entende-se que este trabalho cumpriu o seu papel, realizando todas as etapas necessárias para o alcance do seu objetivo. No que diz respeito à problemática levantada no escopo inicial, conclui-se que o trabalho realizado nesta dissertação conseguiu reunir informações suficientes para respondê-la.

Nas próximas subseções, serão descritas as contribuições e limitações da plataforma Voice Home, além da discussão dos futuros trabalhos relacionados a esta pesquisa.

7.2. CONTRIBUIÇÕES

O principal legado deste trabalho refere-se à construção de um modelo e sua consolidação em uma plataforma para a área de Automação Residencial, com as seguintes contribuições:

- Criar uma plataforma que prevê a integração de diferentes dispositivos eletrônicos em um ambiente residencial, independente do seu modelo e/ou marca.
- Disponibilizar uma plataforma que permita a interação dos usuários de forma fácil e dinâmica pela voz, em português do Brasil.
- Construir uma interface para a plataforma que reconheça os comandos local e em nuvem, quando necessário.
- Desenvolver um *middleware* que identifica automaticamente os dispositivos no ambiente.
- Fornecer uma solução de baixo custo aos potenciais clientes.

7.3. LIMITAÇÕES

Quando se realiza uma pesquisa científica, sempre aparecerão limitações para a solução proposta. Os limites servem como motivação para que novos pesquisadores continuem a investigar e melhorar a solução proposta inicialmente. Dessa maneira, ao utilizar a plataforma percebe-se que ela precisa de pontos de melhoria em alguns aspectos, conforme pode ser visto a seguir:

- Os participantes do experimento foram alunos que não possuíam restrições físicas, seria importante realizar um teste da plataforma com pessoas que tivessem restrições de locomoção.
- O experimento foi realizado com apenas dois atuadores e dois dispositivos eletrônicos, sendo necessária a realização de novos experimentos com uma quantidade maior de atuadores e uma variedade maior de dispositivos eletrônicos.
- A distância média entre os usuários e os microfones deve ser levada em consideração. Desta forma, em novos experimentos, a proposta é

disponibilizar mais microfones para diminuir a distância entre o usuário e os microfones, possibilitando aprimorar a captura dos comandos de voz.

- As aplicações implementadas, devem estar aptas a funcionar em condições de ruído de fundo, desta forma, seria importante realizar experimentos com a utilização de microfones com tecnologia que reduza o ruído de fundo, para verificar possíveis melhorias no sistema.
- Incorporar à plataforma outros tipos de interações como, por exemplo, interações via gestos e interações via dispositivos móveis.

7.4. TRABALHOS FUTUROS

A popularização da área de Automação Residencial está intrinsecamente ligada a evolução da eletrônica e da computação, que vem tornando os dispositivos computacionais cada vez menores e mais rápidos. Como resultado, as soluções que utilizam plataformas com interação via voz podem se tornar mais baratas para a criação de soluções na área de domótica.

Para dar continuidade a este trabalho, faz-se necessário realizar as melhorias citadas na Seção 7.2. Além disso, o teste da plataforma com outros modelos de atuadores aumentará a sua abrangência de utilização. O ambiente também poderia ser capaz de detectar situações como, por exemplo, de perigo e tomar uma ação.

Outro ponto importante a ser destacado é a utilização de outras modalidades de interação, já que esse é um dos principais recursos disponibilizados pelo *middleware*. Isso torna possível que pessoas que tenha problemas físicos ou cognitivos possam utilizar o cenário de interação, mesmo com restrições de visão, audição, fala e coordenação motora. Neste sentido, para avaliar a eficiência e eficácia da plataforma como uma ferramenta de inclusão social, bastaria integrar ao ambiente outros sensores que possibilitem capturar comandos via gestos ou via dispositivos móveis.

Outra perspectiva é incrementar o modelo utilizar outros sensores com o objetivo de alimentar a Camada de Processamento com informações contextuais do ambiente, de forma a aperfeiçoar suas respostas às interações dos usuários.

No que diz respeito aos métodos de avaliação, pretende-se realizar mais experimentos com diferente tipos de pessoas:

- Utilizando maior quantidade de participantes.
- Utilizando técnicas de análise quantitativa mais precisas, tal como, o método estatístico *Analysis of Variance* (ANOVA), no âmbito de tornar os resultados mais confiáveis e aplicáveis à comunidade científica.

Por fim, pretende-se que os resultados encontrados durante a realização desta pesquisa sejam apresentados em trabalhos acadêmicos, tais como: artigos científicos e periódicos. Desta forma, espera-se que o trabalho forneça contribuições que possam ser utilizadas para a melhoria contínua da solução proposta nesta dissertação, contemplando novas expectativas e anseios dos especialistas nas áreas de Automação Residencial e Interação Humano-Computador.

REFERÊNCIAS

AARTS, E.;ENCARNACAO, J. **True Visions**: The Emergence of Ambient Intelligence. Springer: Berlin, German, 2006. 1-16.

AUGUSTO, J.; MCCULLAGH, P. Ambient intelligence: Concepts and applications. **Computer Science and Information Systems/ComSIS**, v.4, n.1, p. 1-26,2007.

ALENCAR, T. S. D.; NERIS, V. P. D. A. Sistemas Ubíquos para Todos: conhecendo e mapeando os diferentes perfis de interação. In: BRAZILIAN SYMPOSIUM ON HUMAN FACTORS IN COMPUTING SYSTEMS,12., Brasília, 2013 **Proceedings …** 2013, p.178-187.

ATZORI, L.; IERA, A.; MORABITO, G. The Internet of Things: a survey. **Computer Networks,** 2010.

ALSHU'EILI, H.; GUPTA, G. S.; MUKHOPADHYAY, S. Voice Recognition Based Wireless Home Automation System. In: INTERNATIONAL CONFERENCE ON MECHATRONICS (ICOM), 4., 2011, 17-19 May 2011, Kuala Lumpur, Malaysia. **Proceedings…** 2011.

BARBOSA, P. A. Máquinas falantes como instrumentos linguísticos: por um humanismo éclairé**. Línguas e Instrumentos Linguísticos**, n. 8, p. 51-99, jul./dez. 2001. Disponível em: <http://www.unicamp.br/iel/site/docentes/plinio/LingInstLing.pdf >. Acesso em: 16 jul. 2014

BLAKE, J. Natural User Interfaces.**Net**. Manning Publications Company, 2011.

BRAJNIK, G. Web Accessibility Testing: When The Method is the Culprit. **Computers Helping People with Special Needs, LNCS,** v. 4061, Springer, p. 156–163, 2006.

BOUAKAZ, S. et al. **CIRDO:** Smart companion for helping elderly to live at home for longer. **IRBM,** lsevier Masson, , v.35, n.2, p.101-108, 2014..

BRUSTOLIN, A. **Itinerário do uso e variação de nós e a gente em textos escritos e orais de alunos do Ensino Fundamental da Rede Publica de FLorianopolis.** Florianopólis: Universidade Federal de Santa Catarina, 2009.

CAVALLO, F. et al. On the design, development and experimentation of the ASTRO assistive robot integrated in smart environments**.** In ROBOTICS AND AUTOMATION (ICRA), IEEE INTERNATIONAL CONFERENCE, 2013. **Proceedings…** 2013. P.4310-4315.

CARICOS. **CARICOS**, 2016. Disponível em: <http://www.caricos.com/cars/v/vw/2013_ volkswagen_golf/1920x1080/98.html>. Acesso em: 2 maio 2016.

CERÓN, I.F.C.; BADILLO, A.G.G. **A Keyword Based Interactive Speech Recognition System for Embedded Applications.** 2011. Master's Thesis, June, 2011.

CHAHUARA, P. et al. On-line Human Activity Recognition from Audio and Home Automation Sensors: comparison of sequential and non-sequential models in realistic Smart Homes**. Journal of ambient intelligence and smart envi-ronments**, 2016, v.8, n.4, p.399-422. Disponível em: <http://content.iospress.com/journals/journal-of-ambient-intelligence-and-smart-environments/8/4>. Acesso em: 2 maio 2016.

COOK, D. J.; AUGUSTO, J. C.; JAKKULA, V. R. Ambient intelligence: Technologies, applications, and opportunities. **Pervasive and Mobile Computing**, v. 8, p.277-298, 2009.

CRUTZEN, C.K.M. Invisibility and the meaning of ambient intelligence. **International Review of Information Ethics,** v**.** 6,p. 1-11, 2006.

CYBENKO, G. Approximation by superposition of sigmoidal functions. **Mathematics of Control, Signals and Systems,** v. 2, n. 4, p. 303-314, 1989.

DAINES, H. ET AL. A. **Pocketsphinx: A Free, Real-Time Continuous Speech Recognition System for Hand-Held Devices**. In: Proc. of the ICASSP, Toulouse, France (2006)

DEY, A.; ABOWD, G.; SALBER, D. **A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications**. Human-Computer Interaction, 16(2), (2001) pp. 97-166.

DUTOIT, T. **An Introduction to Text-To-Speech Synthesis**. Kluwer Academic Publishers, 1997.

DUCATEL, K. ET AL. **Scenarios for Ambient Intelligence** in 2010. [S.l.]: IPTS-Seville, 2001.

EDWARDS, J.; FOREWORD-BY-ORFALI, R. **3 Tier Client/Server at Work.** [S.l.]: John Wiley & Sons, Inc., 1998.

FERGUSON, J. E., **Hidden Markov Models for Speech.** [S.l.]: IDA, Princeton, NJ, 1980.

FIGUEIREDO, L. et al. **Interação Natural a partir de Rastreamento de Mãos**. [S.l.]: [s.n.], 2012.

FLANAGAN, J. L. **Speech Analysis:** synthesis and perception. [S.l.]: Springer-Verlag, 1972.

GAIDA, C. et al. **Comparing open-source speech recognition toolkits, tech. rep., Technical report.** [S.l.]: Project OASIS, 2014.

GAO, X. T. et al. Assist Disabled to Control Electronic Devices and Access Computer Functions by Voice Commands. In: INTERNATIONAL CONVENTION ON

REHABILITATION ENGINEERING & ASSISTIVE TECHNOLOGY: IN CONJUNCTION WITH 1ST TAN TOCK SENG HOSPITAL NEUROREHABILITATION MEETING, 1., New York, NY, USA 2007. **Proceedings**… 2007. p. 37-42.

GOMES, R. J. R. **Teste de interface de voz**. 2007. Dissertação (Mestrado)- Universidade do Porto – Porto, 2007.

GÁRATE, A. et al Ambient Intelligence as paradigm of a full Automation Process at Home in a real application. In: INTERNATIONAL SYMPOSIUM ON COMPUTATIONAL INTELLIGENCE IN ROBOTICS AND AUTOMATION, Espoo, Finland, 2005. **Proceedings**… 2005.

HAMILL, M. et al Development of an automated speech recognition interface for personal emergency response systems**. Journal of NeuroEngineering and Rehabilitation,** v. 6, 2009.

HANSEN, J. **An investigation of smartphone applications: exploring usability aspects related to wireless personal area networks, context-awareness, and remote information access.** 2012. Thesis (Doctor)- School of Information Systems, Computing and Mathematics, Brunel, 2012.

HAYKIN, S. S. **Neural networks:** a comprehensive foundation. Upper Saddle River: Prentice-Hall, 1999.

HEVNER, A.; CHATTERJEE, S. Design research in information systems: theory and practice. **Springer Science & Business Media**, v. 22, 2010.

HIRAMA, K. **Engenharia de software:** qualidade e produtividade com tecnologia. [S.l.]: [s.n.], 2012.

HUO, X.; PARK, H.; KIM, J.; GHOVANLOO, M. A Dual-Mode Human Computer Interface Combining Speech and Tongue Motion for People with Severe Disabilities. **IEEE Transactions on Neural Systems and Rehabilitation Engineering**, v. 21, n. 6, nov. 2013.

IBGE. 2012. **Escolaridade e rendimento aumentam e cai mortalidade infantil**. Disponível em: <http://saladeimprensa.ibge.gov.br/noticias?busca=1&id=1&idnoticia=2125&t=censo-2010 -escolaridade-rendimento-aumentam-cai-mortalidade-infantil&view=noticia> Acesso em: 18 maio 2016.

ISO 9241-11. **Ergonomic requirements for office work with visual display terminals (VDTs)** Part 11: Guidance on Usability. ISO. 2008.

JUFARSKY; MARTIN. **Speech and language processing**. Londres: Pearson International Edition, 2009.

KEMPELEN, W. V. **Mechanismus der menschlichen Sprache nebst der Beschreibung seiner sprechenden Maschine**. Vienna: Degen, J. V., 1970.

KRONBAUER, A. H.; SANTOS, C. A. S. Uma análise das abordagens para avaliar a usabilidade de smartphones: estado da arte e novas tendências. In: SIMPÓSIO BRASILEIRO SOBRE FATORES HUMANOS EM SISTEMAS COMPUTACIONAIS (IHC 2013), 12., Manaus, 2013. **Proceedings**… 2013. P.452-461.

KRONBAUER, A. H.; SANTOS, C. A. S.; VIEIRA, V. Smartphone Applications Usability Evaluation: A Hybrid Model and Its Implementation. Lecture Notes. In: WINCKLER, M.; FORBRIG, P. ; BERNHAUPT, R. (Org.). **Computer Science.** 1. st. ed. 2012. [S.l.]: [s.n.], p.146-163.

KUMAR, A. et al.Rethinking Speech Recognition on Mobile Devices. **IUI4DR,** California, USA.feb. 2011.

LISTERRI, J.; MARTÍ ANTONÍN, M. A. **Tratamiento del lenguaje natural**. Barcelona: Edicions de la Universitat de Barcelona, S.L. Unipersonal, 2002.

LÓPEZ, R.; CALLEJAS, Z. Multimodal dialogue for ambient intelligence and smart environments. In: H. NAKASHIMA, H. ; AGHAJAN, J.C. Augusto (Eds.) **Handbook of Ambient Intelligence and Smart Environments.** [S.l.]: Springer US, 2010. p. 559–579

LECOUTEUX, B.; VACHER, M.; PORTET, F. Distant speech recognition in a smart home: Comparison of several multisource ASRs in realistic conditions. In: INTERSPEECH 2011, Florence, Italy, 2011. p. 2273–2276

LEE, A.; KAWAHARA T.; SHIKANO K. Julius an open source real-time large vocabulary recognition engine. In: EUROPEAN CONFERENCE ON SPEECH COMMUNICATION AND TECHNOLOGY (EUROSPEECH), 2001. **Proc…** 2001. p. 1691-1694.

LIKERT, R. A technique for the measurement of attitudes. **Archives of psychology**, n.140, p.1–55, 1932.

LIU, W. **Natural User Interface – Next Mainstream Product User Interface**. 2010. Disponível em <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5681374> Acesso em: 9 out. 2015.

MARTINS, V. F.; BRASILIANO, A.; FERNANDES, L. F. Interface do usuário baseada em voz como ferramenta para promover o ensino/aprendizagem de língua estrangeira. **REAVI-Revista Eletrônica do Alto Vale do Itajaí**, v. 1, n. 1, p. 34-42, 2012.

MARDIANA, B. et al. Homes appliances controlled using speech recognition in wireless network environment. In: ICCTD 2009 - 2009 INTERNATIONAL CONFERENCE ON COMPUTER TECHNOLOGY AND DEVELOPMENT, FACULTY OF ELECTRONICS AND COMPUTER ENGINEERING, Malacca, Malaysia, 2009. **Proc…** Malacca, Malaysia:Universiti Teknikal Malaysia Melaka, 2009.

MARELI, D. et al. Um framework de desenvolvimento de aplicações ubíquas em ambientes inteligentes. In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUÍDOS, 31., Brasília, DF, 2013. **Anais...** p.643-656.

MAEDA, E.; MINAMI, Y. Steps toward ambient intelligence. **NIT Technical Review,** v.4, n.1, 2006.

MEDEIROS, L. F. **Redes neurais em Delphi.** 2. ed. Florianópolis: Visual Books, 2006. 210 p.

MEGALINDAS. 2016. Disponível em: <http://megalindas.com/decorar-una-sala-inteligente/>. Acesso em: 26 abr. 2016.

MICROSOFT. **Help Microsoft Speech SDK version 5.1.** Microsoft Corporation, 2002.

MINSKY, M.; PAPERT, S. **Perceptrons**. Cambridge, MA: MIT Press, 1969.

MORBINI, F. et al. Which ASR should I choose for my dialogue system? In: SIGDIAL, 2013. **Proceedings**... 2013. p. 394-403.

NASCIMENTO, M. **Os manuais escolares do ensino básico no ensino da linguística.** 2011. Dissertação (Mestrado). Universidade da Beira Interior - Portugal, 2011.

NETO, N. et al. Free tools and resources for Brazilian Portuguese speech recognition. **Journal of the Brazilian Computer Society,** Federal University of Pará, Augusto Correa, 1, Belém, Brazil; IST/INESC-ID, Alves Redol, 9, Lisbon, Portugal, 2011.

NIELSEN, J. Usability inspection methods. In: CONFERENCE COMPANION ON HUMAN FACTORS IN COMPUTING SYSTEMS, 1994. **Proceedings...** 1994. p. 413-414.

OLIVEIRA A.L.C. Brazilian Portuguese speech-driven answering system [Sistema de atendimento com interação de fala para o Português do Brasil]. In: EURO AMERICAN CONFERENCE ON TELEMATICS AND INFORMATION SYSTEMS, EATIS 2012, 6., 2012, Sergipe. **Proceedings...** 2012.

O'HARA, K. et al. On the naturalness of touchless: Putting the “interaction” back into NUI. **ACM Transactions on Computer-Human Interaction (TOCHI)**, v. 20, n. 1, p. 5, 2013.

OLSON, H.F.; BELAR, H. Phonetic Typewriter, **J. Acoust. Soc. Am**., v.28, n.6, p. 1072-1081, 1956.

PEREIRA, M.H.R. et al A multimedia information system to support the discourse analysis of vídeo recordings of television programs. In: IBERIAN CONFERENCE ON INFORMATION SYSTEMS AND TECHNOLOGIES, CISTI, Belo Horizonte, 2012. **Anais...** Belo Horizonte, Minas Gerais, Brazil; Instituto de Ciências Exatas e Biológicas - ICEB, Universidade Federal de Ouro Preto - UFOP, Ouro Preto, Minas Gerais, Brazil, 2012,

PEREIRA, L. A. D. M. **Automação residencial:** rumo a um futuro pleno de novas. São Paulo: [s.n.], 2007.

PEREIRA, L. A.; RAOUFI, M.; FROST, J. C. **Using MySQL and JDBC in new teaching methods for undergraduate database systems courses.** Data Engineering and Management. Springer Berlin Heidelberg, 2012. p.245-248.

PICCOLO, L. S. G.; MENEZES, E. M.; BUCCOLO, B. C. Developing an Accessible Interaction Model for Touch Screen Mobile Devices: Preliminary Results. In: SYMPOSIUM ON HUMAN FACTORS IN COMPUTING SYSTEMS, 10., LATIN AMERICAN CONFERENCE ON HUMAN-COMPUTER INTERACTION, 5., ACM,2011. **Proc….** 2011.

PHAM, P.; PHAM, A. **Scrum em Ação–Gerenciamento e Desenvolvimento Ágil de Projetos de Software**.São Paulo: Novatec, 2011.

PICCOLO, L. S. G.; MENEZES, E. M.; BUCCOLO, B. C. Developing an Accessible Interaction Model for Touch Screen Mobile Devices: Preliminary Results. In: SYMPOSIUM ON HUMAN FACTORS IN COMPUTING SYSTEMS, 10., LATIN AMERICAN CONFERENCE ON HUMAN-COMPUTER INTERACTION, 5., ACM,2011. **Proc….** 2011.

PORTET, F. et al. Design and evaluation of a smart home voice interface for the elderly: acceptability and objection aspects. **Personal and Ubiquitous Computing**, v.17, n.1, p.127-144, 2014.

POZA, M.; VILLARRUBIA, L.; SÁNCHES, J. Teoría y aplicaciones de reconocimiento automático del habla. **Comunicaciones de Telefónica, Investigacion y Desarrolo**, n.3, jun. 1991.

PREECE, J.; SHARP, H.; ROGERS, Y. **Interaction Design-beyond human-computer interaction.** [S.l.]: John Wiley & Sons, 2015.

QUESENBERY, W. What does usability mean: Looking beyond 'ease of use, In: 48th ANNUAL CONFERENCE SOCIETY FOR TECHNICAL COMMUNICATION, 48., Chicago, 2001.**Proc….** 2001.

RAJABZADEH, A.; MANASHTY, A. R.; JAHROMI, Z. F. **A Mobile Application for a Smart House Remote Control System**. World Academy of Science, Engineering and Technology, Kermanshah, 2010.

RAMANATHAN, R.; KORTE, T. Software service architecture to access weather data using RESTful web services. **Computing, Communication and Networking Technologies**, 2014, 1-8.

RABINER, L., JUANG, B.H. **Fundamentals of speech recognition**, Prentice Hall, 1993.

RABINER, L. **A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition**, Proceedings of the IEEE, vol.77, no.2, pp.257–86, Feb.1989.

RODRIGUES, R.; MACIEL, A.; FILHO, E. Desenvolvimento de uma ferramenta para a produção de mídias utilizando personagem animado com síntese de voz. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE 2012), 23., Rio de Janeiro, 2012. **Anais...** 2012.

SABERELETRONICA. Eletrônica. **sabereletronica**, 2016. Disponível em: <http://www.sabereletronica.com.br/artigos/1733-tecnologias-de-redes-de-comunicao-para-sistemas-automotivo>. Acesso em: 26 abr. 2016.

SANTAELLA, L. et al. Desvelando a Internet das Coisas. **Revista GEMInIS,** v. 1, n. 2, p. 19-32, 2013.

SATRIA, A. et al. The Framework of Home Remote Automation System Based on Smartphone**. International Journal of Smart Home**, Jakarta, 2015.

SANTOS, M. **Interface Multimodal de Interação Humano-Computador em Sistema de Recuperação de Informação Baseado em Voz e Texto em Português**. 2013. Dissertação (Mestrado)- Universidade de Brasília - UNEB, Brasil, 2013.

SCHLOGL, S. et al. **Exploring Voice User Interfaces for Seniors.** Proceedings of the 6th International Conference on PErvasive Technologies Related to Assistive Environments. Article No. 52 , May 29-31 2013, Island of Rhodes, Greece

SILVA, P. et al. **An open-source speech recognizer for Brazilian Portuguese with a windows programming interface.,** Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Signal Processing Laboratory, Universidade Federal do Pará, 2010.

SILVA, D. **Algoritmos de processamento da linguagem natural para sistemas de conversão texto-fala em português**. 2008. Tese (Doutorado) – Departamento de Galego-Portugués, Francés e Lingüística, Universidade da Coruña, Coruña, 2008.

SINGER, T. Tudo conectado: conceitos e representações da internet das coisas. In: SIMPÓSIO EM TECNOLOGIAS DIGITAIS E SOCIABILIDADE, 10., 2012. **Anais...** 2012.

SOUZA, J. A. **Reconhecimento de padrões usando indexação recursiva.** Universidade Federais de Santa Catarina, 1999.

SPHINX, C. **Version of decoders.** Disponível em: <http://cmusphinx.sourceforge.net/wiki/ versions> 2012. Acesso em: 2 mar. 2015.

SHARMA, A.; KUMAR, A.; BHARDAWAJ, A. **International Journal of Information & Computation Technology,** v.4, n. 9, p. 879-8842014.

SHARP, H.; ROGERS, Y.; PREECE. J. Interaction Design: Beyond Human-Computer Interaction. **3nd New York: John Wiley & Sons**. 2011, ISBN: 978-0470665763.

TAYLOR, P. **Text-to-Speech Synthesis**. Cambridge: Cambridge University Press 2007. Disponível em: <http://mi.eng.cam.ac.uk/~pat40/book.html>. Acesso em: 8 jul. 2011.

UNIFACS. 2014. Disponível em: <http://www.gmr.unifacs.br/lab/labengenharias.php?guia=1&&lab=13>. Acesso em: 2 maio 2016.

VALLI, A. **Natural Interaction White Paper.** [S.l.]: Citeseer, 2007. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.98.9153\&rep=rep1\&type=pdf.

VASILAKOS, A.; PEDRYCZ, W. **Ambient Intelligence, Wireless Networking, and Ubiquitous Computing**. [S.l.]: Artech House Publishers, 2006.

VACHER, M. et al.The SWEET-HOME Project: Audio Technology in Smart Homes to improve Well-being and Reliance. In: ANNUAL INTERNATIONAL CONFERENCE OF THE IEEE EMBS, 33., Boston, Massachusetts USA, 2011. **Proceedings**… August 30 - September 3, 2011.

VACHER, M. et al. **The Sweet-Home speech and multimodal corpus for home automation interaction.** The 9th edition of the Language Resources and Evaluation Conference (LREC), Reykjavik: Iceland (2014).

VACHER, M. et al **Ac-quisition et reconnaissance automatique d'expressions et d'appels vocaux dans un habitat**. JEP-TALN-RECITAL 2016, Jul 2016, Paris, France. p.28-36, 2016.

VACHER, M.; LECOUTEUX, B.; PORTET, F. **On Distant Speech Recognition for Home Automation.** Lecture Notes in Computer Science, 8700, Springer, pp.161-188, 2015, Smart Health: Open Problems and Future Challenges, 978-3-319-16225-6.

VACHER, M. et al. Evaluation of a context-aware voice interface for Ambient Assisted Living: qualitative user study vs. quantitative system evaluation. **ACM - Transactions on Speech and Language Processing, Association for Computing Machinery, Special Issue on Speech and Language Processing for AT (Part 3),** v.7, n.2, p.5:1-5:36, 2015.

VETERE, F. et al. NUI: social perspectives in natural user interfaces. In: COMPANION PUBLICATION ON DESIGNING INTERACTIVE SYSTEMS. ACM, 2014. **Proceedings**… 2014..p. 215-218.

WALSH, V.; TAYLOR H.R. Autoxatic speech recoghltioh usihg artificial irtelligehce kethods. In: EUROPEAN CONFERENCE ON SPEECH TECHNOLOGY EDINBURGH, 1987, Scotland, UK. **Proceedings**… 1987.

WEISER, M. The computer for the 21st century. **Scientific American**, p. 66-75, 1991.

WEIS, T. et al. Rapid prototyping for pervasive applications. **Pervasive Computing,** IEEE, v.6, p. 76-84, 2007.

WEISS, B. et al. Quality of talking heads in different interaction and media contexts**. Speech Communication,** v.52, p.) 481–492, 2010.

WIGDOR, D.; WIXON, D. **Brave NUI world:** designing natural user interfaces for touch and gesture. Morgan Kaufmann, 2011.

WILLIAMS, R. **Real-Time Systems Development**. [S.l.]: Butterworth-Heinemann. 1st ed. 2006.

HELAL, S. et al. The gator tech smart house: a programmable pervasive space. **Computer,** v.38, n.3, p. 50-60, 2005.

HUANG, X.; ACERO, A.; HON H. **Spoken language processing**. [S.l.]: Prentice-Hall, 2001.

XBOX. **Kinect**. Disponível em: <http://www.xbox.com/pt-BR/Kinect/Home-new>. Acesso em: 7 jan. 2016.

YI, J.Z. et al Microcontroller Based Voice-Activated Powered Wheelchair Control. In: INTERNATIONAL CONVENTION ON REHABILITATION ENGINEERING & ASSISTIVE TECHNOLOGY: IN CONJUNCTION WITH 1ST TAN TOCK SENG HOSPITAL NEUROREHABILITATION MEETING, 2007, New York, NY, USA. **Proceedings**… 2007.

YOUNG, S. et al **The HTK book.** Cambridge university engineering department 3. [S.l.]: [s.n.], 2002. p. 175.

YU, Y. Research on speech recognition technology and its application, Proceedings – 201. 2 In: INTERNATIONAL CONFERENCE ON COMPUTER SCIENCE AND ELECTRONICS ENGINEERING, ICCSEE 2012, Putian, Fujian, China. **Proceedings**… Putian, Fujian, China: Department of Electronics and Information Engineering, Putian University, 2012.

ANEXO A

UNIVERSIDADE SALVADOR - UNIFACS/BA

PARECER CONSUBSTANCIADO DO CEP

DADOS DO PROJETO DE PESQUISA

Título da Pesquisa: UMA PLATAFORMA PARA INTERAÇÕES NATURAIS EM AMBIENTES INTELIGENTES

Pesquisador: Artur Henrique Kronbauer

Versão: 1

CAAE: 51137215.4.0000.5033

Instituição Proponente: Universidade Salvador - UNIFACS/BA

Patrocinador Principal: Financiamento Próprio

DADOS DO PARECER

Número do Parecer: 1.366.396

Apresentação do Projeto:

O projeto intitulado "Uma plataforma para interações naturais em ambientes inteligentes" contextualiza a computação ubíqua enquanto um mecanismo que facilite a interação entre usuários e dispositivos computacionais de modo que os comandos disparados pelos usuários sejam os mais naturais possíveis. E que para proporcionar esta funcionalidade, os sistemas ubíquos deverão capturar informações sobre o ambiente para dinamicamente se adaptar ao contexto e automaticamente executar ações apropriadas a cada mudança no cenário de interação.

Objetivo da Pesquisa:

O objetivo geral deste projeto é pesquisar e desenvolver alternativas tecnológicas para a criação de uma plataforma multimodal, possibilitando a interação de pessoas com sistemas computacionais de forma natural, utilizando gestos, voz e toques em dispositivos móveis.Já os objetivos específicos são: - Criar um middleware que identifique automaticamente os sensores (equipamentos para captura de imagens, sons e dispositivos móveis), os atuadores (microcontroladores que possam acionar funcionalidades em dispositivos eletrônicos) e possa fazer a comunicação entre eles.

- Construir uma infraestrutura para o tratamento de imagens, sons e acionamentos remotos via dispositivos móveis, permitindo a interpretação de comandos executados pelos usuários no cenário de interação.

UNIVERSIDADE SALVADOR - UNIFACS/BA

Continuação do Parecer: 1.366.396

Avaliação dos Riscos e Benefícios:

O questionário refere-se apenas a critérios de usabilidade e desempenho da plataforma, não se utilizando de métodos invasivos ou que exponha as participantes, dessa forma não se observa riscos aparentes para o conjunto amostral da pesquisa. No tocante aos benefícios, após o desenvolvimento do projeto será possível identificar se a plataforma proposta contempla índices de usabilidade aceitáveis, tem a potencialidade de ajudar na inclusão social e pode auxiliar a vida das pessoas em tarefas do cotidiano.

Comentários e Considerações sobre a Pesquisa:

Com o progresso dos sistemas embarcados e das tecnologias que utilizam redes sem fio, percebe-se o avanço na área de Ambientes Inteligentes e a evolução dos estudos de interações entre os seres humanos e dispositivos eletrônicos por meio de ações naturais. Este cenário favorece o desenvolvimento da computação ubíqua que está pautada na idéia da computação estar presente em todos os locais e ser transparente aos seres humanos. Assim como a evolução dos motores, que hoje estão presentes na maioria dos artefatos que cercam as pessoas, mas são imperceptíveis, os sistemas computacionais devem seguir o mesmo caminho e tornarem-se onipresentes.

Considerações sobre os Termos de apresentação obrigatória:

O projeto apresenta todos os termos de apresentação obrigatória.

Recomendações:

Sem recomendações.

Conclusões ou Pendências e Lista de Inadequações:

Projeto eticamente apto para desenvolvimento.

Considerações Finais a critério do CEP:

Mantido parecer do relator.

Situação do Parecer:

Aprovado

Necessita Apreciação da CONEP:

Não

SALVADOR, 14 de Dezembro de 2015

ANEXO B

Termo de Consentimento Livre e Esclarecido

Comitê de Ética em Pesquisa da UNIFACS
Programa de Pós-Graduação em Sistemas e Computação (PPGCOMP)
Laureate International Universities
Avenida Luís Viana Filho, Nº 3146, Imbuí - CEP: 41720-200 - Salvador-Bahia.
Telefone (71) 3021-2800.

Uma Plataforma para Interações Naturais Via Voz em Ambiente Inteligente

Eu _____;
estou sendo convidado (a) a participar de um experimento para avaliar a usabilidade e aspectos práticos relacionados a utilização de uma plataforma de interação baseada em interações naturais via gestos em ambiente inteligente, desenvolvida para facilitar a interação das pessoas com dispositivos eletrônicos em ambientes domésticos ou empresariais.

A plataforma foi concebida como projeto de pesquisa do professor Dr. Artur Henrique Kronbauer, vinculado ao Programa de Pós-Graduação em Sistemas Computacionais (PPGCOMP) da Universidade Salvador. O referido pesquisador pode ser contatado pelo telefone (71) 99111-8409 ou pelo endereço de e-mail arturhk@gmai.com.

Recebi esclarecimentos sobre a pesquisa e estou ciente de que minha privacidade será respeitada, ou seja, meu nome e imagem serão mantidos em sigilo. Eu autorizo a utilização do questionário contendo as minhas respostas preenchidas durante a realização do experimento, entendo que as informações serão utilizadas somente para os fins desta pesquisa, bem como serão divulgadas apenas em artigos e na redação dos trabalhos científicos orientados pelo professor Dr. Artur Henrique Kronbauer.

Estou ciente que poderei solicitar esclarecimentos quanto a quaisquer dúvidas durante a realização do experimento e terei acesso aos resultados obtidos. Tenho ciência de que poderei me recusar a responder qualquer pergunta e que posso me negar a participar do estudo ou retirar meu consentimento a qualquer momento, sem prévia justificativa.

Manifesto meu livre consentimento em participar.

Salvador, _____ de _____ _____.

_____

Nome e assinatura do participante

_____

Nome e assinatura do pesquisador

ANEXO C

Roteiro para utilização da plataforma

Programa de Pós-Graduação em Sistemas e Computação (PPGCOMP)
Laureate International Universities
Avenida Luís Viana Filho, Nº 3146, Imbuí - CEP: 41720-200 - Salvador-Bahia.
Telefone (71) 3021-2800.

Voice Home: Interações Naturais Via Voz em Ambientes Inteligentes

O experimento será realizado nas dependências da UNIFACS, no período de 04/09 até 05/09. Os usuários serão orientados pelo aluno do Mestrado de Sistemas e Computação Italo Ribeiro Costa dos Santos sobre os procedimentos de utilização das interfaces para realizar o experimento.

O experimento será composto de:

- 1 Microfone;
- 1 dispositivo central (*middleware*);
- 2 atuadores;
- 2 eletrodomésticos (televisão e ar condicionado);

O roteiro do experimento está composto pelas seguintes etapas:

1) Orientação preliminar aos usuários.
2) Assinatura do Termo de Consentimento Livre e Esclarecido.
3) Realização das tarefas com interação natural através de voz em um ambiente inteligente, que serão realizadas da seguinte forma:
    a. Escolherqual eletrodomestico será utilizado na interação
    b. O usuário deve fica na frente do eletrodomestico escolhido e com o microfone realizar o tipo de interação desejada.
    c. Ainteração com a voz deve ser realizada conforme tabela abaixo

Exemplo de Interação	**Ação Resultante Prevista**
“Ligar o ar-condicionado”	Liga o ar-condicionado.
“Desligar o ar-	Desliga o ar-condicionado.

condicionado”	
“Ligar a TV”	Liga a TV.
“Desligar a TV”	Desliga a TV.
“Proximo Canal”	Muda para o próximo canal da TV.
“Canal Anterior”	Muda para o canal anterior da TV.
“Mute”	Coloca o modo mute na TV.
“Aumentar o volume”	Aumenta o volume da TV.
“Dimunuir o volume”	Diminui o volume da TV.

a. O usuário deverá acionar os aparelhos eletrônicos, sendo de livre escolha a sequência de interação.
b. Para finalizar, o usuário deve solicitar ao orientador que terminou de interagir com o ambiente.

4) Após a realização das tarefas, os usuários irão responder a um questionário.
5) Os usuários serão dispensados do experimento quando terminarem de responder ao questionário.

ANEXO D

Avaliação da Plataforma

Programa de Pós-Graduação em Sistemas e Computação (PPGCOMP)
Laureate International Universities
Avenida Luís Viana Filho, Nº 3146, Imbuí - CEP: 41720-200 - Salvador-Bahia.
Telefone (71) 3021-2800.

Questionário Proposto aos Participantes do Experimento

Local: _____ Data: ___/___/___

Parte 1 - Dados Pessoais

1) Nome:_____
___

2) Idade:
_____

3) Curso de Graduação:
_____

4) Ano de Formação:
_____

5) Atividade Profissional:_____

6) Empresa onde trabalha:
_____

7) Sexo: ( ) Masculino ( ) Feminino

Parte 2 - Avaliação da Usabilidade com a Plataforma de Interação Natural via Voz

8) Qual o seu nível de satisfação com a rapidez (eficiência) com que consegue realizar as tarefas através da voice home?

	Muito Satisfeito
	Satisfeito

	Indiferente
	Insatisfeito
	Muito Insatisfeito

9) Qual o seu nível de satisfação com a precisão (eficácia) com que consegue executar as tarefas desejadas através da voice home?

	Muito Satisfeito
	Satisfeito
	Indiferente
	Insatisfeito
	Muito Insatisfeito

10) Qual é seu nível de satisfação com a utilização da voice home?

	Muito Satisfeito
	Satisfeito
	Indiferente
	Insatisfeito
	Muito Insatisfeito

11) Qual o seu nível de satisfação com a aprendizagem das funcionalidades da voice home?

	Muito Satisfeito
	Satisfeito
	Indiferente
	Insatisfeito
	Muito Insatisfeito

12) Quando houve uma interação indesejada, você conseguiu facilmente retornar para um estado anterior com o objetivo de iniciar a tarefa novamente?

	Muito Fácil
	Fácil

	Médio
	Difícil
	Muito Difícil

13) Qual o seu nível de satisfação com a acessibilidade disponibilizada pela voice home?

	Muito Satisfeito
	Fácil
	Médio
	Difícil
	Muito Difícil

14) Qual o seu nível de satisfação com relação à flexibilidade (caminhos alternativos para executar uma tarefa) disponíveis na voice home?

	Muito Satisfeito
	Satisfeito
	Indiferente
	Insatisfeito
	Muito Insatisfeito

15) Qual o seu nível de satisfação com a utilidade da voice home?

	Muito Satisfeito
	Satisfeito
	Indiferente
	Insatisfeito
	Muito Insatisfeito

16) Qual a sua percepção com relação a facilidade de uso da voice home?

	Muito Fácil
	Fácil
	Médio
	Difícil

	Muito Difícil

17) Você tem alguma consideração que gostaria de relatar para aprimorar o desenvolvimento da plataforma proposta?

ANEXO E

CÓDIGO-FONTE DO *MIDDLEWARE*

Classe Atuador

```
package com.bean;

public class Atuador {
	private int id;
	private String mac;
	private String nome;
	private String ip;
	private int status;

	public Atuador(){
		setId(0);
		setMac("");
		setNome("");
		setStatus(0);
	}

	public Atuador(int id, String mac, String nome, int status)
	{
		setId(id);
		setMac(mac);
		setNome(nome);
		setStatus(status);
	}

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getMac() {
		return mac;
	}
	public void setMac(String mac) {
		this.mac = mac;
	}
	public int getStatus() {
		return status;
	}
	public void setStatus(int status) {
		this.status = status;
	}

	public String getNome() {
		return nome;
	}

	public void setNome(String nome) {
		this.nome = nome;
	}
	public String getIp() {
```

```
		return ip;
	}

	public void setIp(String ip) {
		this.ip = ip;
	}

}
```

Classe Controller

```
package com.bean;

import java.util.ArrayList;
import java.util.Date;

public class Controller {
	private int id;
	private String nome;
	private Date dataAcesso;
	private String tipo;
	private ArrayList<Atuador> atuadores;

	public Controller(){
		id = 0;
		nome = "";
		dataAcesso = new Date(System.currentTimeMillis());
		tipo = "";
		atuadores = new ArrayList<Atuador>();
	}

	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getNome() {
		return nome;
	}
	public void setNome(String nome) {
		this.nome = nome;
```

```
}
public Date getDataAcesso() {
	return dataAcesso;
}
public void setDataAcesso(Date data) {
	this.dataAcesso = data;
}
public String getTipo() {
	return tipo;
}
public void setTipo(String tipo) {
	this.tipo = tipo;
}
public ArrayList<Atuador> getAtuadores() {
	return atuadores;
}

public Atuador getAtuador(int pos) {
	return atuadores.get(pos);
}

public void setAtuadores(ArrayList<Atuador> atuadores) {
	this.atuadores = atuadores;
}

public void addAtuadores(Atuador atuador) {
	this.atuadores.add(atuador);
}

public Atuador buscaAtuadores(Atuador a){

	for (int i = 0; i < this.getAtuadores().size(); i++) {
		if (a.getId() == this.getAtuador(i).getId()){
			a.setIp(this.getAtuador(i).getIp());
			this.getAtuador(i).setStatus(a.getStatus());
			return a;
		}
	}
	a.setStatus(123);
	return a;
```

```
    }
    }
```

Classe JSONParser

```
package com.raspberry;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONException;
import org.json.JSONObject;

@SuppressWarnings("deprecation")
public class JSONParser
{
static InputStream is = null;
static JSONObject jObj = null;
static String json = "";

public JSONObject getJSONFromUrl(String url)
{
   // Requisição HTTP
   try {
      // defaultHttpClient

    DefaultHttpClient httpClient = new DefaultHttpClient(); // instancia o objeto httpClient para receber a URL do atuador ou raspbe
      HttpPost httpPost = new HttpPost(url);

      HttpResponse httpResponse = httpClient.execute(httpPost);
      HttpEntity httpEntity = httpResponse.getEntity();
      is = httpEntity.getContent();
```

```
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    } catch (ClientProtocolException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }

    try {
        BufferedReader reader = new BufferedReader(new InputStreamReader(
                is, "iso-8859-1"), 8);
        StringBuilder sb = new StringBuilder();
        String line = null;
        while ((line = reader.readLine()) != null) {
            sb.append(line + "\n");
            System.out.println(line); //imprime todo o Objeto Json da url
        }
        is.close();
        json = sb.toString();
    } catch (Exception e) {
    }

    // try parse the string to a JSON object
    try {
        jObj = new JSONObject(json);
    }
    catch (JSONException e) {

        System.out.println("error on parse data in jsonparser.java");
    }

    // retorno da String do JSON
    return jObj;
}
}
```

Classe ListaWebServerHttpAtuador

```
package com.raspberry;

import java.io.IOException;
```

```
import java.net.HttpURLConnection;
import java.net.URL;
import org.json.JSONException;
import org.json.JSONObject;
import com.bean.Atuador;

public class ListaWebServerHttpAtuador
{
static int contador =0;

	static public Atuador  ping (final String address, int tempo) throws IOException
	{

		Atuador atuador = new Atuador();
			try {

				final URL url = new URL("http://" + address+"/T");
				final HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
				urlConn.setConnectTimeout(tempo); // Tempo de reposta em ms enviado por parâmetro
				urlConn.connect();

				if (urlConn.getResponseCode() == HttpURLConnection.HTTP_OK)
				{
					System.out.println("Equipamento encontrado em: " + address);
					//System.out.println("Ping em "+address +" Sucesso!");
						contador++;
						atuador = leituraAtuador("http://"+address+"/T"); //leitura é a função dessa classe que chama a classe do jsonparse
						atuador.setIp(address);
					return atuador;
					//fazer função para testar por json se é o atuador e guardar o nome, ip, cômodo, qtd de equipamentos de cada host encontrado(atuadores);
				}

				else
				{
					System.out.println("Falhou: " + address);
					return null;
```

```
                }
                }

                catch (Exception e)
                {
                        //System.err.println("Ping FALHOU: " + address + " - " +
                        e);
                        return null;
                }


} // fim da função PING

public static Atuador leituraAtuador (String url) throws JSONException
{
        String conteudo;
        JSONObject object;
        JSONParser jsonParser;
        Atuador atuador = new Atuador();

        try
        {

                jsonParser = new JSONParser();
                System.out.println(url);
                System.out.println("passo 01");
                object = jsonParser.getJSONFromUrl (url);
                System.out.println("passo 02");
                conteudo = object.getString("id");
                System.out.println("passo 03");
                atuador.setId(object.getInt("id"));
                atuador.setNome(object.getString("nome"));

                atuador.setStatus(Integer.parseInt(object.getString("lamp")));

                System.out.println("Json encontrado, valor de parametro chave: "+
                conteudo);
                return atuador;
        }
        catch(Exception e)
        {
```

```
                    System.out.println("Json não encontrado");
                    return null;
                }
        } // fim da função leitura.


} // fim da classe
```

Classe PostHttp

```
package com.raspberry;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLConnection;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;

@SuppressWarnings("deprecation")
public class PostHttp
{

    private DefaultHttpClient httpClient;

    public void postt(String url2)
    {



        try {
                /*
                 final URL url = new URL(url2);
```

```
                final HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
                urlConn.setConnectTimeout(1000); // Tempo de reposta em ms enviado por parâmetro
                 urlConn.connect();

              if (urlConn.getResponseCode() == HttpURLConnection.HTTP_OK)
              {
                     System.out.println(" post ok : ");
              }

              else
              {
                     System.out.println("Falhou post : ");
              }
            }
          */

                  URL url = new URL(url2);
        URLConnection yc = url.openConnection();
        BufferedReader in = new BufferedReader(new InputStreamReader(yc.getInputStream()));
        String inputLine;

        while ((inputLine = in.readLine()) != null)
           System.out.println(inputLine);

        in.close();
           }

             catch (Exception e)
             {
                     //System.err.println("Ping FALHOU: " + address + " - " + e);

       }

           /*
           try {

                  httpClient = new DefaultHttpClient();
        HttpPost httpPost = new HttpPost(url);
```

```
            HttpResponse httpResponse = httpClient.execute(httpPost);
            HttpEntity httpEntity = httpResponse.getEntity();
                } catch (Exception e)
                  {
                            System.out.println("Falha no Post");
                           //System.err.println("Ping FALHOU: " + address + " - " + e);

               }

                  /*try {
          // defaultHttpClient

         httpClient = new DefaultHttpClient();
          HttpPost httpPost = new HttpPost(url);

          HttpResponse httpResponse = httpClient.execute(httpPost);
          HttpEntity httpEntity = httpResponse.getEntity();









      } catch (UnsupportedEncodingException e) {
          e.printStackTrace();
      } catch (ClientProtocolException e) {
          e.printStackTrace();
      } catch (IOException e) {
          e.printStackTrace();
      }
      */
        }
}
```

Classe ServidorHttp

```
package com.raspberry;

import java.io.IOException;
```

```
import java.io.OutputStream;
import java.net.InetAddress;
import java.net.InetSocketAddress;
import java.net.URI;
import java.net.UnknownHostException;
import java.util.HashMap;
import java.util.Map;
import com.bean.Atuador;
import com.bean.Controller;
import com.context.ConfigurationHandHandler;
import com.sun.net.httpserver.HttpExchange;
import com.sun.net.httpserver.HttpHandler;
import com.sun.net.httpserver.HttpServer;

public class ServidorHttp
{
   static Controller sensor;
   static int contador=0;
   static int temp = 20;

   public static void main(String[] args) throws Exception
  {
        sensor = new Controller();
        encontra();
        ConfigurationHandHandler configuration = new ConfigurationHandHandler(sensor);
        HttpServer server = HttpServer.create(new InetSocketAddress(8080), 0);
        server.createContext("/info", new InfoHandler());
        server.createContext("/get", new GetHandler());
        server.createContext("/configurationHand", configuration);
        server.setExecutor(null); // creates a default executor
        server.start();
        System.out.println("O servidor foi iniciado");
  } // Fim do main

  // http://localhost:8000/info
  static class InfoHandler implements HttpHandler
  {
    public void handle(HttpExchange httpExchange) throws IOException
    {
      //Buscar os atuadores
```

```
    //encontra();
    //imprimir json com todos os atuadores dinamicamente com o ip, tipo, estado e o id a partir da classe Sensor
    String response = consultaAtuadores();

    ServidorHttp.writeResponse(httpExchange, response.toString());
    System.out.println("Válido info");
    return;
  }
 }

 static class GetHandler implements HttpHandler
 {
  public void handle(HttpExchange httpExchange) throws IOException
  {
    //String response = "Olá, o Servidor está online em http://localhost/get?atuador=12&acao=123";
    String response = "[{status:ok}]";
    String acaoAtuador = "";
    //Obter os Parametros
    URI requestedUri = httpExchange.getRequestURI();
     String query = requestedUri.getRawQuery();
     Map<String, String> parameters = ServidorHttp.queryToMap(query);

    System.out.println("Obter atuador:" + parameters.get("atuador") + " Ação: " +parameters.get("acao"));

    Atuador atuadorResq = new Atuador(Integer.parseInt(parameters.get("atuador")), "", "", Integer.parseInt(parameters.get("acao")));
    PostHttp obj = new PostHttp();

    atuadorResq = sensor.buscaAtuadores(atuadorResq);

    //Ligar LED
    if (atuadorResq.getStatus() == 0)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/L"; // Ligar LED
    else if(atuadorResq.getStatus() == 1)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/H"; // Desligar LED

    //Ligar Ar
```

```
    else if(atuadorResq.getStatus() == 4)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/H"; // Ligar ar
    else if(atuadorResq.getStatus() == 5)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/L"; // Desligar ar
    //Mudar Temperatura do Ar
    else if(atuadorResq.getStatus() == 6){
            temp++;
            if (temp > 21)
                    temp = 21;
            acaoAtuador = "http://"+atuadorResq.getIp()+"/"+ String.valueOf(temp); //Aumenta
Temperatura
    }else if(atuadorResq.getStatus() == 7){
            temp--;
            if (temp < 18)
                    temp = 18;
            acaoAtuador = "http://"+atuadorResq.getIp()+"/"+ String.valueOf(temp); //Diminui
Temperatura
    }


    else if(atuadorResq.getStatus() == 15)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/m"; // Louver


    //Ligar TV
    else if(atuadorResq.getStatus() == 2)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/P"; // Ligar ou Desligar TV
    else if(atuadorResq.getStatus() == 3)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/R"; // Desligar TV - Verificar se é
necessário
    else if(atuadorResq.getStatus() == 13)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/i"; // Source - Verificar se é necessário
    //Mudar Canal da TV
    else if(atuadorResq.getStatus() == 9)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/c"; //Mudar Canal TV
    else if(atuadorResq.getStatus() == 10)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/cm"; // Diminuir Canal TV
    //Aumentar Volume da TV
    else if(atuadorResq.getStatus() == 11)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/v"; // Aumenta Volume da TV
    else if(atuadorResq.getStatus() == 12)
            acaoAtuador = "http://"+atuadorResq.getIp()+"/vm"; // Diminui Volume da TV
    else if(atuadorResq.getStatus() == 14)
```

```
                acaoAtuador = "http://"+atuadorResq.getIp()+"/mu"; // Mute da TV
    obj.postt(acaoAtuador);

    ServidorHttp.writeResponse(httpExchange, response);
     System.out.println("Válido get");
  }
 }

 public static void writeResponse(HttpExchange httpExchange, String response) throws
IOException
 {
  httpExchange.sendResponseHeaders(200, response.length());
  OutputStream os = httpExchange.getResponseBody();
  os.write(response.getBytes());
  os.close();
 }

 public static Map<String, String> queryToMap(String query){
  Map<String, String> result = new HashMap<String, String>();
  if (query != null){
       for (String param : query.split("&")) {
          String pair[] = param.split("=");
          if (pair.length>1) {
             result.put(pair[0], pair[1]);
          }else{
             result.put(pair[0], "");
          }
       }
  }
  return result;
 }

 public static void encontra() throws UnknownHostException
 {
     System.out.println("IP/Localhost: " + InetAddress.getLocalHost().getHostAddress());
              //String ip = InetAddress.getLocalHost().getHostAddress();
        String ip = "192.168.0.1";

          String oct[] = ip.split("\\.");
                      String hostAtuadores;
```

```
                int  tempo;
                tempo = 50;

                for (int i = 0; i < 255; i++)
                {
                hostAtuadores = oct[0] + "." + oct[1] + "." + oct[2] + ".";
                hostAtuadores = hostAtuadores + String.valueOf(i);

                Atuador atuador;
                try {
                        if(Integer.parseInt(oct[3]) != i)
                        {
                                atuador = ListaWebServerHttpAtuador.ping(hostAtuadores,
tempo);
                                if (atuador != null){

                                        //Inserir no Arraylist de atuadores
                                        sensor.addAtuadores(atuador);

                                        System.out.println("Equipamento encontrado em: " +
hostAtuadores);

                                        // obj.leitura(hostAtuadores);
                                        contador++;
//fazer função para testar por json se é o atuador e guardar o nome, ip, cômodo,qtd de
equipamentos de cada host encontrado(atuadores);
                                    }else
                                    {
                                        System.out.println("Falhou: " + hostAtuadores);
                                    }
                        }
                 }catch (Exception e)
                  {
                                System.err.println("Ping FALHOU: " + hostAtuadores + " - " +
e);
                    }

                }

                System.out.println("O Número de web servers é: " + contador);
        // if(contador==0)
```

```
                //encontra();
    }

    public static String consultaAtuadores(){
        String retorno = "";

        retorno =  "[";
        for (int i = 0; i < sensor.getAtuadores().size(); i++) {

                sensor.getAtuador(i);

                if(i > 0)
         {
            retorno = retorno + ",";
         }

          retorno = retorno + "{";
          retorno = retorno + " \"id\":\""+sensor.getAtuador(i).getId()+"\", ";
          retorno = retorno + " \"ip\":\""+sensor.getAtuador(i).getIp()+"\", ";
          retorno = retorno + " \"mac\":\""+sensor.getAtuador(i).getMac()+"\", ";
          retorno = retorno + " \"nome\": \""+ sensor.getAtuador(i).getNome() + "\", ";
          retorno = retorno + " \"status\": \"" + sensor.getAtuador(i).getStatus() + "\" " ;
          retorno = retorno + "}\n ";
        }
        retorno = retorno + "]";
        return retorno;
     }
    }
```

CÓDIGO-FONTE DO VOICE HOME

Classe Translator

```
import javax.swing.JButton;

import com.bean.Controller;

import ConectionPost.PostHttp;
import ConectionPost.Postt;

//Esta classe foi criada para transformar os comandos recebidos em suas devidas aÃ§Ãµes
public class Translator {
        private static PostHttp post = new PostHttp();
```

```
        private static Postt post2 = new Postt();
        public static Controller c = new Controller();
          public static String translate(String s){


                  if(s.equals("ligar")==true){


                          System.out.println("Executar post para ligar ");
                          post2.postt("http://192.168.0.119/P"); //TV
                          //post2.postt("http://192.168.0.113/get?atuador="+ "2" + "&acao="+
"2"); //TV
                         // post2.postt("http://192.168.0.116/get?atuador="+ "1" + "&acao="+
"2"); //TV

                          /*
                                                                          Obtem os
atuadores
                          c = post.obterAtuadores("localhost", 500);
                          if (c.getAtuador(0).getStatus() == 4){
                                                Veirifica se ja esta ligado
                                  System.out.println("O arc-condicionado se encontra ligado ");
                          //}else {

                                 post.postAcao("http://localhost/get?atuador="+ "1" + "&acao="+
"4");
                           }
                          */

                          return "ligar";

              }
              if(s.equals("desligar")==true){

                 System.out.println("Executar post para desligar ");
                 post2.postt("http://192.168.0.119/P"); //TV
                 //post.postAcao("http://192.168.0.116/get?atuador="+ "1" + "&acao="+ "3");
//TV

                                  /*
                                  c = post.obterAtuadores("localhost", 500);
                                  if (c.getAtuador(0).getStatus() == 5){
                                          System.out.println("O arc-condicionado se encontra
desligado ");
                                  //}else {

                                        post.postAcao("http://localhost/get?atuador="+ "1" +
"&acao="+ "5");
                                   }
                                  */

                 return "desligar";
              }
```

```
            if(s.equals("proximo")==true || s.equals("mudar")==true ||
s.equals("prÃ³ximo")==true){

                System.out.println("Executar post para mudar ");
                post2.postt("http://192.168.0.119/c"); //TV canal
                //post.postAcao("http://192.168.0.116/get?atuador="+ "1" + "&acao="+ "3");
//TV

                                        /*
                                        c = post.obterAtuadores("localhost", 500);
                                        if (c.getAtuador(0).getStatus() == 5){
                                                System.out.println("O arc-condicionado se encontra
desligado ");
                                        //}else {

                                                post.postAcao("http://localhost/get?atuador="+ "1" +
"&acao="+ "5");
                                         }
                                        */

                return "mudar";
            }

            if(s.equals("anterior")==true){

                System.out.println("Executar post para anterior ");
                post2.postt("http://192.168.0.119/cm"); //TV canal
                //post.postAcao("http://192.168.0.116/get?atuador="+ "1" + "&acao="+ "3");
//TV

                                        /*
                                        c = post.obterAtuadores("localhost", 500);
                                        if (c.getAtuador(0).getStatus() == 5){
                                                System.out.println("O arc-condicionado se encontra
desligado ");
                                        //}else {

                                                post.postAcao("http://localhost/get?atuador="+ "1" +
"&acao="+ "5");
                                         }
                                        */

                return "anterior";
            }

            if(s.equals("aumentar")==true){

                System.out.println("Executar post para aumentar ");
                post2.postt("http://192.168.0.119/v"); //TV volume
                //post.postAcao("http://192.168.0.116/get?atuador="+ "1" + "&acao="+ "3");
//TV

                                        /*
                                        c = post.obterAtuadores("localhost", 500);
                                        if (c.getAtuador(0).getStatus() == 5){
```

```
                                        System.out.println("O arc-condicionado se encontra
desligado ");
                                    //}else {

                                        post.postAcao("http://localhost/get?atuador="+ "1" +
"&acao="+ "5");
                                     }
                                    */

                    return "aumentar";
                }

                if(s.equals("diminuir")==true){

                    System.out.println("Executar post para diminuir ");
                    post2.postt("http://192.168.0.119/vm"); //TV volume
                    //post.postAcao("http://192.168.0.116/get?atuador="+ "1" + "&acao="+ "3");
//TV

                                    /*
                                    c = post.obterAtuadores("localhost", 500);
                                    if (c.getAtuador(0).getStatus() == 5){
                                        System.out.println("O arc-condicionado se encontra
desligado ");
                                    //}else {

                                        post.postAcao("http://localhost/get?atuador="+ "1" +
"&acao="+ "5");
                                     }
                                    */

                    return "diminuir";
                }

                if(s.equals("mute")==true){

                    System.out.println("Executar post para mute ");
                    post2.postt("http://192.168.0.119/mu"); //TV
                    //post.postAcao("http://192.168.0.116/get?atuador="+ "1" + "&acao="+ "3");
//TV

                                    /*
                                    c = post.obterAtuadores("localhost", 500);
                                    if (c.getAtuador(0).getStatus() == 5){
                                        System.out.println("O arc-condicionado se encontra
desligado ");
                                    //}else {

                                        post.postAcao("http://localhost/get?atuador="+ "1" +
"&acao="+ "5");
                                     }
                                    */

                    return "mute";
                }
```

```
                        return null;
                                                                    //Se nÃ£o reocnhecer
devolve null



}
}
```

Classe RecoListener

```
import javax.speech.recognition.Result;
import javax.speech.recognition.ResultAdapter;
import javax.speech.recognition.ResultEvent;
import javax.speech.recognition.ResultToken;
import javax.swing.JButton;
import com.nuvem.speech.SyncRecognizeClient;

//Essa classe Ã© o Listener do reconhecedor que fica sempre esperando algum comando
de voz
//A voz processada sera enviada a classe Translator.java.
public class RecoListener extends ResultAdapter{
        public void resultAccepted(ResultEvent e) {
                String Recognized = null;
                String path = "/home/italo/workspace/ProjetoVoz/Dic.grammar";
                String bt;
                try {
                        Result r = (Result) (e.getSource());
                        ResultToken tokens[] = r.getBestTokens();
                        for (int i = 0; i < tokens.length; i++){

                                Recognized = tokens[i].getSpokenText();
                        }
                        bt = Translator.translate(Recognized);
//                      bt = Apresenta.translate(Recognized);

                        //Se nÃ£o for reconhecida localmente usa a Nuvem
                        if (bt == null) {
                                String txt = "--host=speech.googleapis.com --port=443 --
uri=e.getDir() --sampling=16000";
                        String[] argsx = txt.split(" ");
                        SyncRecognizeClient.main(argsx);
                        //A classe reconhece e devolve para o Translator
                           }

            ManipuladorGramatical.insereGramatica(bt,path);
            //Atualiza o texto que foi reconhecido no Translator

                }catch (Exception e1) {
                        e1.printStackTrace();
                }
        }

}
```

Classe Recognize (reconhecimento local)

```
import java.io.FileReader;
import java.io.IOException;

import javax.speech.AudioException;
import javax.speech.Central;
import javax.speech.EngineException;
import javax.speech.EngineStateError;
import javax.speech.recognition.DictationGrammar;
import javax.speech.recognition.GrammarException;
import javax.speech.recognition.Recognizer;
import javax.speech.recognition.RecognizerModeDesc;
import javax.speech.recognition.ResultAdapter;
import javax.speech.recognition.RuleGrammar;

public class Recognize extends ResultAdapter {

	public static Recognizer rec;
	public static RuleGrammar gram, gram2;
	public static DictationGrammar dic;

	public static void main() {

		try {

			RecognizerModeDesc rmd = (RecognizerModeDesc) Central
				.availableRecognizers(null).firstElement();

			rec = Central.createRecognizer(rmd);// cria o reconhecedor
			System.out.println("rec created");
			String dic_grammar = null;
			dic_grammar = BuscaGramatica.retorno();

			rec.allocate();//carrega os arquivos do Projeto Coruja
			FileReader reader = new
FileReader(System.getProperty("user.home")+dic_grammar);
			// faz a leitura do arquivo de gramÃ¡tica de comando e controle
			gram = rec.loadJSGF(reader);
			//importa os modelos acusticos e de linguagem para que o
reconhecimento seja em Portugues do Brasil
			dic = rec.getDictationGrammar("dicSr");

			dic.setEnabled(false);

			gram.addResultListener(new RecoListener());
			//cria o listener para a variÃ¡vel da gramÃ¡tica
			rec.resume();
			//inicia o reconhecedor
			System.out.println("rec resume");

		} catch (IOException e) {
```

```
                    e.printStackTrace();
                } catch (IllegalArgumentException e ) {
                    e.printStackTrace();
                } catch (SecurityException e) {
                    e.printStackTrace();
                } catch (EngineStateError e) {
                    e.printStackTrace();
                } catch (AudioException e) {
                    e.printStackTrace();
                } catch (EngineException e) {
                    e.printStackTrace();
                } catch (GrammarException e) {
                    e.printStackTrace();
                }
        }


}
```

Classe ManipuladorGramatical

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class ManipuladorGramatical {

    public static void leitor(String path) throws IOException {
        BufferedReader buffRead = new BufferedReader(new FileReader(path));
        String linha = "";
        while (true) {
            if (linha != null) {
                System.out.println(linha);

            } else
                break;
            linha = buffRead.readLine();
        }
        buffRead.close();
    }

    public static void escritor(String path) throws IOException {
        BufferedWriter buffWrite = new BufferedWriter(new FileWriter(path));
        String linha = "";
        Scanner in = new Scanner(System.in);
        System.out.println("Escreva algo: ");
        linha = in.nextLine();
```

```
        buffWrite.append(linha + "\n");
        buffWrite.close();
    }

    public static void insereGramatica(String gramaticaNova, String path) throws IOException
{

        BufferedReader buffRead = new BufferedReader(new FileReader(path));
        String linha = "";
        String linha2 = null;
        while (true) {
            if (linha != null) {

                    if (linha2 == null) {

                            if (linha != "") {
                                    linha2 = linha;
                            }

                    }else
                            linha2 = linha2 + "\n" + linha;

            } else
                break;
            linha = buffRead.readLine();

        }

        if (linha2.contains(gramaticaNova)) {
            buffRead.close();
        }else
            {buffRead.close();
            linha2 = linha2.replace(";", " | ");
            linha2 = linha2 + gramaticaNova + ";";
             BufferedWriter buffWrite = new BufferedWriter(new FileWriter(path));
             buffWrite.append(linha2 + "\n");
             buffWrite.close();
            }

    }

}
```

Classe SyncRecognizeClient (reconhecimento nuvem)

```
package com.nuvem.speech;

import com.examples.cloud.speech.AsyncRecognizeClient;
import com.examples.cloud.speech.RecognitionAudioFactory;
import com.google.api.client.*;
import com.google.auth.Credentials;
import com.google.auth.oauth2.GoogleCredentials;
import com.google.cloud.speech.v1beta1.RecognitionAudio;
```

```
import com.google.cloud.speech.v1beta1.RecognitionConfig;
import com.google.cloud.speech.v1beta1.RecognitionConfig.AudioEncoding;
import com.google.cloud.speech.v1beta1.SpeechGrpc;
import com.google.cloud.speech.v1beta1.SyncRecognizeRequest;
import com.google.cloud.speech.v1beta1.SyncRecognizeResponse;
import com.google.protobuf.TextFormat;

import io.grpc.ManagedChannel;
import io.grpc.StatusRuntimeException;

import org.apache.commons.cli.CommandLine;
import org.apache.commons.cli.CommandLineParser;
import org.apache.commons.cli.DefaultParser;
import org.apache.commons.cli.Option;
import org.apache.commons.cli.Options;
import org.apache.commons.cli.ParseException;

import java.io.IOException;
import java.net.URI;
import java.util.Arrays;
import java.util.Collection;
import java.util.Collections;
import java.util.List;
import java.util.concurrent.TimeUnit;
import java.util.logging.Level;
import java.util.logging.Logger;

import org.apache.commons.cli.*;

/**
 * Client that sends audio to Speech.SyncRecognize and returns transcript.
 */
public class SyncRecognizeClient {

  private static final Logger logger =
Logger.getLogger(SyncRecognizeClient.class.getName());

  private static final List<String> OAUTH2_SCOPES =
      Arrays.asList("https://www.googleapis.com/auth/cloud-platform");

  private final URI input;
  private final int samplingRate;

  private final ManagedChannel channel;
  private final SpeechGrpc.SpeechBlockingStub speechClient;

  /**
   * Construct client connecting to Cloud Speech server at {@code host:port}.
   */
  public SyncRecognizeClient(ManagedChannel channel, URI input, int samplingRate)
      throws IOException {
    this.input = input;
    this.samplingRate = samplingRate;
    this.channel = channel;
```

```
  speechClient = SpeechGrpc.newBlockingStub(channel);
}

private RecognitionAudio createRecognitionAudio() throws IOException {
  return RecognitionAudioFactory.createRecognitionAudio(this.input);
}

public void shutdown() throws InterruptedException {
  channel.shutdown().awaitTermination(5, TimeUnit.SECONDS);
}

/** Send a non-streaming-recognize request to server. */
public void recognize() {
  RecognitionAudio audio;
  try {
    audio = createRecognitionAudio();
  } catch (IOException e) {
    logger.log(Level.WARNING, "Failed to read audio uri input: " + input);
    return;
  }
  logger.info("Sending " + audio.getContent().size() + " bytes from audio uri input: " + input);
  RecognitionConfig config =
      RecognitionConfig.newBuilder()
          .setEncoding(AudioEncoding.LINEAR16)
          .setSampleRate(samplingRate)
          .setLanguageCode("pt-BR")
          .build();
  SyncRecognizeRequest request =
      SyncRecognizeRequest.newBuilder().setConfig(config).setAudio(audio).build();

  SyncRecognizeResponse response;
  try {
    response = speechClient.syncRecognize(request);
  } catch (StatusRuntimeException e) {
    logger.log(Level.WARNING, "RPC failed: {0}", e.getStatus());
    return;
  }
  logger.info("Received response: " + TextFormat.printToString(response));
}

public static void main(String[] args) throws Exception {

  String audioFile = "resources/audio.raw";
  String host = "speech.googleapis.com";
  Integer port = 443;
  Integer sampling = 16000;

  CommandLineParser parser = new DefaultParser();

  Options options = new Options();
  options.addOption(
      Option.builder()
          .longOpt("uri")
          .desc("path to audio uri")
          .hasArg()
```

```
            .argName("FILE_PATH")
            .build());
    options.addOption(
        Option.builder()
            .longOpt("host")
            .desc("endpoint for api, e.g. speech.googleapis.com")
            .hasArg()
            .argName("ENDPOINT")
            .build());
    options.addOption(
        Option.builder()
            .longOpt("port")
            .desc("SSL port, usually 443")
            .hasArg()
            .argName("PORT")
            .build());
    options.addOption(
        Option.builder()
            .longOpt("sampling")
            .desc("Sampling Rate, i.e. 16000")
            .hasArg()
            .argName("RATE")
            .build());

    try {
      CommandLine line = parser.parse(options, args);


      if (line.hasOption("host")) {
        host = line.getOptionValue("host");
      } else {
        System.err.println("An API enpoint must be specified (typically
speech.googleapis.com).");
       System.exit(1);
      }

      if (line.hasOption("uri")) {
          audioFile = line.getOptionValue("uri");
        } else {
          System.err.println("An Audio uri must be specified (e.g. file:///foo/baz.raw).");
         System.exit(1);
        }

      if (line.hasOption("port")) {
        port = Integer.parseInt(line.getOptionValue("port"));
      } else {
        System.err.println("An SSL port must be specified (typically 443).");
        System.exit(1);
      }

      if (line.hasOption("sampling")) {
        sampling = Integer.parseInt(line.getOptionValue("sampling"));
      } else {
        System.err.println("An Audio sampling rate must be specified.");
        System.exit(1);
```

```
      }
    } catch (ParseException exp) {
      System.err.println("Unexpected exception:" + exp.getMessage());
      System.exit(1);
    }

    ManagedChannel channel = AsyncRecognizeClient.createChannel(host, port);
    SyncRecognizeClient client = new SyncRecognizeClient(channel, URI.create(audioFile),
sampling);
    try {
      client.recognize();
    } finally {
      client.shutdown();
    }
  }
}
```

CÓDIGO-FONTE DOS ATUADORES

Ar condicionado

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <IRremoteESP8266.h>

IRsend irsend(2);

const char* ssid     = "AML";
const char* password = "raspberry";
ESP8266WebServer server(80);
String webString="";

void handle_root()
{
  server.send(200, "text/plain", "Hello from the esp8266, read from /T or /H and /L ");
  delay(100);
}

void setup(void)
{
  irsend.begin();
  Serial.begin(115200);
  pinMode(2, OUTPUT);
```

```
 // Conexão com a Wi-Fi
 WiFi.begin(ssid, password);
 Serial.print("\n\r \n\rWorking to connect");

 // Wait for connection
 while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.print(".");
 }
 Serial.println("");
 Serial.print("Connected to ");
 Serial.println(ssid);
 Serial.print("IP address: ");
 Serial.println(WiFi.localIP());

 server.on("/", handle_root);

 server.on("/T", []()
 {  // Informações do Atuador

   webString="{\"lamp\":\"00\",\"id\":\"03\",\"nome\":\"Ar\"}";
   server.send(200, "text/plain", webString);
 });

 server.on("/H", []()
 {
   unsigned int  irSignal1[] = {8850,4500, 500,600, 600,500, 550,1650, 550,600, 600,1600, 650,500,
550,550, 550,600, 600,500, 550,1700, 550,550, 600,500, 600,550, 550,550, 550,1700, 600,1600,
550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 600,500, 650,500, 550,550,
550,600, 550,550, 550,600, 550,550, 550,600, 550,600, 550,550, 550,550, 550,600, 550,550,
550,600, 550,550, 550,1700, 550,600, 550,1650, 550,1650, 550,600, 550,550, 550};
   webString="GPIO2 esta enviado";
   int khz = 38; // 38kHz frequencia usada no protocolo NEC
   irsend.sendRaw(irSignal1, sizeof(irSignal1) / sizeof(irSignal1[0]), khz);
   delay(40);
   server.send(200, "text/plain", webString);
 });

 server.on("/L", []()
 {
   unsigned int  irSignal[] = {8850,4450, 550,550, 600,550, 550,1650, 550,600, 550,550, 550,600,
550,550, 550,600, 550,550, 550,1700, 550,550, 550,600, 550,550, 550,600, 550,1650, 550,1650,
```

```
550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,600, 550,550, 550,550,
600,550, 550,550, 550,600, 550,550, 550,600, 550,600, 550,550, 550,600, 550,550, 550,600,
550,550, 550,550, 600,1650, 550,600, 550,550, 550,1650, 550,600, 550,550, 550};
   webString="GPIO2 esta enviado";
   int khz = 38; // 38kHz frequencia usada no protocolo NEC
   irsend.sendRaw(irSignal, sizeof(irSignal) / sizeof(irSignal[0]), khz);
   delay(40);
   server.send(200, "text/plain", webString);
 });


   server.on("/m", []()
 {
  unsigned int  irSignal2[] = {8900,4450, 550,550, 650,500, 550,1650, 550,550, 550,1700, 550,550,
650,500, 550,1650, 550,600, 550,1650, 550,550, 550,600, 550,1650, 550,600, 550,1650, 550,600,
550,550, 550,600, 600,500, 550,600, 550,550, 550,600, 550,550, 600,550, 600,500, 550,600,
550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 600,550, 550,550, 550,600,
550,550, 550,600, 550,1650, 550,600, 550,600, 550,1650, 550,550, 550,550, 600};
webString="GPIO2 esta enviado";
   int khz = 38; // 38kHz frequencia usada no protocolo NEC
   irsend.sendRaw(irSignal2, sizeof(irSignal2) / sizeof(irSignal2[0]), khz);
   delay(40);
   server.send(200, "text/plain", webString);
 });


   server.on("/18", []()
 {
   unsigned int  irSignal10[] = {8900,4450, 600,500, 600,500, 550,1700, 550,550, 550,1650, 550,600,
550,550, 650,500, 550,600, 550,1650, 550,550, 550,600, 550,550, 550,600, 600,1600, 550,1700,
550,550, 550,600, 550,550, 550,600, 550,550, 550,550, 550,600, 550,600, 550,550, 550,600,
550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600,
550,550, 550,600, 550,1650, 550,600, 550,1650, 550,1650, 600,550, 550,550, 550};
   webString="GPIO2 esta enviado";
   int khz = 38; // 38kHz frequencia usada no protocolo NEC
   irsend.sendRaw(irSignal10, sizeof(irSignal10) / sizeof(irSignal10[0]), khz);
   delay(40);
   server.send(200, "text/plain", webString);
 });


    server.on("/19", []()
 {
   unsigned int  irSignal4[] = {8900,4450, 500,600, 550,550, 550,1650, 600,550, 550,1650, 550,600,
550,550, 550,600, 550,550, 550,1700, 550,550, 550,600, 550,550, 550,1650, 550,600, 550,600,
550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600,
550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600,
550,550, 550,600, 550,1650, 550,600, 550,1650, 550,600, 550,1650, 550,1650, 550};
```

```
    webString="GPIO2 esta enviado";
    int khz = 38; // 38kHz frequencia usada no protocolo NEC
    irsend.sendRaw(irSignal4, sizeof(irSignal4) / sizeof(irSignal4[0]), khz);
    delay(40);
    server.send(200, "text/plain", webString);
  });


     server.on("/20", []()
  {
    unsigned int  irSignal5[] = {8900,4450, 500,600, 550,550, 550,1700, 550,550, 550,1650, 550,600, 550,550, 550,600, 550,600, 550,1650, 550,550, 550,600, 550,550, 550,1700, 550,550, 550,1700, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 550,550, 550,600, 600,1600, 550,600, 550,1650, 550,600, 550,1650, 550,550, 550};
    webString="GPIO2 esta enviado";
    int khz = 38; // 38kHz frequencia usada no protocolo NEC
    irsend.sendRaw(irSignal5, sizeof(irSignal5) / sizeof(irSignal5[0]), khz);
    delay(40);
    server.send(200, "text/plain", webString);
  });


   server.on("/21", []()
  {
    webString="GPIO2 esta enviado";
     irsend.sendNEC(0x28460000, 32);
    delay(40);
    server.send(200, "text/plain", webString);
  });


  server.begin();
  Serial.println("HTTP server started");
}

void loop(void)
{
  server.handleClient();
}
```

Televisão

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
```

```
#include <ESP8266WebServer.h>
#include <IRremoteESP8266.h>

IRsend irsend(2);

const char* ssid     = "AML";
const char* password = "raspberry";

ESP8266WebServer server(80);

String webString="";
void handle_root()
{
  server.send(200, "text/plain", "Hello from the esp8266, read from /T or /H and /L ");
  delay(100);
}

void setup(void)
{
  irsend.begin();
  Serial.begin(115200);
  pinMode(2, OUTPUT);

  // Conexão com Wi-Fi
  WiFi.begin(ssid, password);
  Serial.print("\n\r \n\rWorking to connect");

  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  server.on("/", handle_root);
```

```
 server.on("/T", []()
 {  Informações do Atuador

   webString="{\"lamp\":\"00\",\"id\":\"02\",\"nome\":\"TV\"}";   // Microcontroler has a hard time with float
to string
   server.send(200, "text/plain", webString);            // send to someones browser when asked
 });

 server.on("/H", []()
 {
   webString="GPIO2 esta ligada";
   digitalWrite(2, LOW);
   server.send(200, "text/plain", webString);              // send to someones browser when asked
 });

 server.on("/L", []()
 {
   webString="GPIO2 esta desligada";
   digitalWrite(2, HIGH);
   server.send(200, "text/plain", webString);              // send to someones browser when asked
 });

   server.on("/P", []()
 {
   webString="GPIO2 esta enviado";

    for(int i=0; i<3;i++)
   {
   irsend.sendSony(0xa90, 12);
   delay(40);
   }
   server.send(200, "text/plain", webString);              // send to someones browser when asked
 });

  server.on("/i", []()
 {
   webString="GPIO2 esta enviado";
   Serial.println("Sony");
   for(int i=0; i<3;i++)
   {
```

```
  irsend.sendSony(0xa50, 12);
  delay(40);
  }
  server.send(200, "text/plain", webString);            // send to someones browser when asked
});

server.on("/v", []()
{
  webString="GPIO2 esta enviado";
  Serial.println("Sony");
   for(int i=0; i<3;i++)
  {
  irsend.sendSony(0x490, 12);
  delay(40);
  }
  server.send(200, "text/plain", webString);            // send to someones browser when asked
});

server.on("/vm", []()
{
  webString="GPIO2 esta enviado";
  Serial.println("Sony");
   for(int i=0; i<3;i++)
  {
  irsend.sendSony(0xC90, 12);
  delay(40);
  }
  server.send(200, "text/plain", webString);            // send to someones browser when asked
});

 server.on("/vm", []()
{
  webString="GPIO2 esta enviado";
  Serial.println("Sony");
   for(int i=0; i<3;i++)
  {
  irsend.sendSony(0xC90, 12);
  delay(40);
  }
  server.send(200, "text/plain", webString);            // send to someones browser when asked
```

```
});

server.on("/c", []()
{
  webString="GPIO2 esta enviado";
  Serial.println("Sony");
   for(int i=0; i<3;i++)
  {
  irsend.sendSony(0x90, 12);
  delay(40);
  }
  server.send(200, "text/plain", webString);            // send to someones browser when asked
});


server.on("/cm", []()
{
  webString="GPIO2 esta enviado";
  Serial.println("Sony");
   for(int i=0; i<3;i++)
  {
  irsend.sendSony(0x890, 12);
  delay(40);
  }
  server.send(200, "text/plain", webString);            // send to someones browser when asked
});

  server.on("/mu", []()
{
  webString="GPIO2 esta enviado";
  Serial.println("Sony");
   for(int i=0; i<3;i++)
  {
  irsend.sendSony(0x290, 12);
  delay(40);
  }
  server.send(200, "text/plain", webString);            // send to someones browser when asked
});
server.begin();
Serial.println("HTTP server started");
```

```
}

void loop(void)
{
  server.handleClient();
}
```