Escola Politécnica da Universidade de São paulo

PMR3500- Trabalho de Conclusão de Curso II em Engenharia Mecatrônica
MONOGRAFIA

Módulo de Robô Sociável para Automação Residencial com Criação de Rotinas por Voz

Alunos: Felipe Hideki Ogawa – 8610381
Luiz Motta da Silva – 8041713

Orientador: Professor Dr. Marcos R. P. Barretto

São Paulo – junho, 2020

SUMÁRIO

Este relatório é apresentado como requisito parcial para obtenção do grau de Engenheiro Politécnico na Escola Politécnica da Universidade de São Paulo. É o produto do nosso próprio trabalho, exceto onde indicado no texto. O relatório pode ser livremente copiado e distribuído desde que a fonte seja citada

RESUMO

O projeto consiste na conceituação e criação em um módulo de robô sociável cuja aplicação é a automação residencial, que permite ao usuário através da interação humano-máquina por voz gerenciar e controlar diversos aparelhos e eletrodomésticos simultaneamente. Para isso, será utilizado algumas tecnologias pré-existentes, como frameworks open-source e softwares de utilização gratuita para que, através da análise desse trabalho, outros projetos semelhantes possam ser desenvolvidos mais facilmente.

Esse deverá ser capaz de reconhecer a fala do usuário em língua portuguesa, transformá-la em texto para que a informação possa ser tratada e interpretada pelo sistema que deverá entender corretamente o comando do usuário e controlar os eletrodomésticos corretos.

Por se tratar de um módulo, o sistema deve possibilitar sua expansão futura. Neste trabalho o foco será no desenvolvimento de um meio de o usuário criar padrões por comandos de voz, que durante o texto são chamados de rotinas. Uma rotina consiste em conjunto de tarefas executadas pelo sistema apenas com um único comando de voz do usuário. Após a rotina criada o usuário pode requisitar ao sistema a sua execução pela voz e o sistema deverá executar todas as tarefas correspondentes à rotina.

Assim, até o final do projeto, será produzida a documentação com todas as informações relativas ao estudo, modelagem e desenvolvimento da solução do problema em questão. Também é parte do projeto criar o módulo de robô sociável que entenda corretamente o comando do usuário dado na língua, crie rotinas e padrões através de comandos de voz, controle eletrodomésticos, responda corretamente ao usuário, seja adaptável para ter compatibilidade com algum protocolo de comunicação utilizado em automação residencial presente no mercado, tenha clareza e fluidez na comunicação e seja rápido e de fácil utilização.

PALAVRAS-CHAVE E FRASES-CHAVE

Automação residencial; Rotinas; Gerenciadores de diálogos; Robô sociável; Língua portuguesa.

1. INTRODUÇÃO

Em relatório de junho de 2019 o Fórum Econômico Mundial (WEF) posicionou robôs sociáveis na segunda colocação entre as dez principais tecnologias que estão surgindo [26]. Estima-se que as vendas de robôs de consumo alcançaram 5,6 bilhões de Dólares em 2018 e espera-se que até o fim de 2025 as vendas cheguem a 19 bilhões de Dólares com mais de 65 milhões de robôs vendidos no ano. Porém qualquer robô não é considerado um robô sociável. Robôs sociáveis são robôs projetados para trabalharem ao lado de, e/ou interagirem a nível social com humanos. Como qualquer outro robô eles utilizam de inteligência artificial para decidir como reagir às informações recebidas por suas câmeras e sensores.

Apesar de não tão noticiados quanto outros avanços tecnológicos, o desenvolvimento de robôs sociáveis tem avançado bastante nas últimas décadas. Hoje em dia os robôs conseguem executar tarefas mais úteis no dia a dia e complexas, e suas capacidades de interação com humanos evoluem cada dia mais. É esperado para os próximos anos que robôs sociais se tornem cada vez mais sofisticados e presentes em nossa sociedade.

Avanços no campo de inteligência artificial como o machine learning são parte importante e propiciaram a base para os avanços atingidos em robôs sociáveis. Esses avanços permitiram que fossem desenvolvidos robôs cujos algoritmos reconhecem voz, faces e emoções, interpretam a fala e os gestos, respondem de maneira adequada a entradas complexas verbais e não verbais, e que se adaptam às necessidades pessoais aprendendo por feedbacks, recompensas e críticas.

Desta forma robôs sociais têm assumido uma grande variedade de funções antes ocupadas por humanos, e a expectativa é que novas formas de aplicação de robôs sociais continuem a surgir. Atualmente robôs sociais já existem, e outros muitos estão sendo desenvolvidos, e suas principais aplicações são nas áreas educacional, de assistência médica, de assistência social e de uso doméstico.

Uma notícia da revista Forbes de janeiro de 2020 coloca o uso de robôs para auxiliar nas tarefas da casa entre as 5 principais tendências para o mercado de automação residencial [28]. Por enquanto os robôs para automação residencial mais acessíveis aos consumidores são os robôs de serviço, como robôs aspiradores de pó e cortadores de grama. Porém, graças aos avanços no campo de inteligência artificial, não demorará para robôs que executem tarefas mais complexas estejam presentes no mercado. Os robôs domésticos prometem um futuro onde tarefas diárias de cuidados com a casa são feitas por máquinas. Para a população de idade mais avançada, ou com necessidades especiais, os robôs sociáveis domésticos também representarão uma maior segurança para elas, podendo chamar por ajuda em caso de algum acidente, ou ajudando a pessoa a se locomover por exemplo.

1.1. MOTIVAÇÃO

Enquanto os robôs sociais ainda não ocuparam o estado da arte da automação residencial, esse campo por enquanto é preenchido por assistentes pessoais que utilizam dos avanços tecnológicos na área de Internet das Coisas (Internet of Things – IoT). Nos últimos anos nós nos acostumamos à cada vez mais conectar os aparelhos de uso diário de nossas casas à internet e a outros aparelhos para tornar nossas vidas mais confortáveis, econômicas, seguras e divertidas. A pesquisa realizada pela Adobe Digital Insights [17] indica um aumento de 103% das vendas de assistentes de voz ao comparar o último trimestre de 2016 com o de 2017 nos Estados Unidos, indicando uma clara tendência dos consumidores americanos a adotarem os assistentes em suas residências. Segundo matéria publicada pela Forbes [28], o valor do mercado de automação residencial deve crescer de 55 bilhões de Dólares em 2016 para 174 bilhões de Dólares em 2025, ou seja, essa tendência deve continuar pelos próximos anos.

Grandes empresas como Amazon, Google, Apple e Microsoft estão focadas em desenvolver e aprimorar seus produtos para esse segmento. Seus assistentes residenciais, funcionam como centrais onde o usuário conecta todos os dispositivos smarts de sua residência. Assim ele consegue controlar todos os dispositivos conectados via um único aplicativo em seu smartphone ou dispositivo. Alguns sistemas ainda permitem ao usuário criar rotinas que acionem mais de um dispositivo e realizem mais de uma tarefa com apenas um comando do usuário. Porém estas tecnologias ainda são recentes e estão em constante desenvolvimento, e, portanto, ainda há bastante espaço para pesquisa e desenvolvimento sobre o assunto.

Atualmente, por existirem diversas plataformas que proveem o serviço de automação residencia e não existir uma padronização para esse tipo de serviço, fabricantes de dispositivos smarts e os consumidores sofrem com problemas de compatibilidade entre os dispositivos e as plataformas. Fabricantes encontram dificuldades para escolher quais plataformas seu produto terá compatibilidade, em busca de permitir que seu produto atinja ao maior número de consumidores possível. Por outro lado, consumidores muitas vezes encontram-se presos à plataforma escolhida e podem não encontrar dispositivos que atendam às suas necessidades compatíveis com plataforma que possuí. Alguns podem optar por utilizar mais de uma plataforma para realizar a automatização desejada, mas além do incomodo de necessitar utilizar uma série de diferentes aplicativos para instalar e controlar os dispositivos incompatíveis também podem ocorrer falhas de segurança decorrentes dessa opção.

Porém o problema da compatibilidade já foi notado pelas principais empresas do segmento. Amazon, Apple e Google anunciaram unirem esforços para criarem uma padronização para a automação residencial que a torne mais simples e segura, que será código aberto e acessível para todos. O projeto conta com a ajuda de diversas outras empresas da área e busca garantir que qualquer dispositivo com tecnologia para automação residencial presente no mercado seja compatível com qualquer assistente residencial presente no mercado, independentemente de quem seja seus fabricantes.

O padrão a ser desenvolvido vai utilizar tecnologias de conexão já existentes e definirá tecnologias que os dispositivos smarts devem suportar. Uma vez a padronização

pronta e disponível a todos, o ambiente de automação residencial será mais simples e seguro, logo mais atrativo para novos consumidores. Assim quando a padronização estiver disponível, ela poderá impulsionar o aumento do crescimento do mercado de automação residencial.

Atualmente os principais assistentes focados na automação residencial presentes no mercado requerem o uso de plataformas, como aplicativos ou interfaces próprias para o computador, para que o usuário consiga instanciar novas rotinas no sistema. Isso requer uma maior intimidade com tecnologias, e um maior esforço e conhecimento de programação por parte do usuário apenas para fazer algo que poderia ser feito de forma mais simples pelo diálogo com a máquina. Assim o diferencial do sistema proposto neste projeto é a possibilidade de o usuário criar rotinas através do diálogo via voz com o sistema. Permitindo ao usuário que crie rotinas de maneira mais simples e intuitiva.

Então neste trabalho serão aprofundados os assuntos de internet das coisas, reconhecimento de voz, sistemas de diálogo humano-máquina, sistemas de controle para smart homes e diversas outras tecnologias que cercam o assunto. Portanto a necessidade de se apreender novas tecnologias, que atualmente estão em alta no mercado, e a direta ligação do projeto com automação definem a base motivacional necessária que justifica o desenvolvimento deste projeto.

1.2. OBJETIVO

O objetivo ao final deste projeto além de aprofundar o conhecimento sobre as tecnologias anteriormente citadas é o desenvolvimento de um módulo de diálogo, para robô sociável, voltado à automação residencial que permita a criação de rotinas por comandos de voz. Durante o projeto o modulo foi conceituado, estruturado, desenvolvido e testado quanto à sua capacidade de controlar dispositivos e criar rotinas. Objetiva-se que o módulo seja de fácil utilização e crie e execute rotinas do usuário apenas pelo diálogo humano-máquina.

1.3. ESTRUTURA DO TRABALHO

Desta forma, este trabalho encontra-se estruturado da seguinte maneira. No segundo capítulo é apresentado o estado da arte das tecnologias disponíveis de assistentes pessoais e de sistemas de diálogo humano máquina. São analisadas as funcionalidades que eles apresentam e o como eles são arquitetados. No terceiro capítulo são apresentadas as tecnologias utilizadas para o desenvolvimento deste projeto. O quarto capítulo é sobre os requisitos de projeto. Nele definimos os requisitos para o desenvolvimento do módulo e quais funcionalidades não serão implementadas. No quinto capítulo é desenvolvida a solução adotada para a criação do módulo. Neste

capítulo é tratado como foi arquitetado, o funcionamento e como as tecnologias do capítulo 3 são utilizadas no módulo desenvolvido.

O penúltimo capítulo traz os resultados obtidos com o protótipo que foi desenvolvido. São testados alguns casos gerais de uso, com possíveis comandos do usuário, e é exibida a resposta apresentada pelo sistema, se condizente ou não com o esperado. O último capítulo é a conclusão deste trabalho onde é analisado se os requisitos do projeto foram atingidos e discutido quais os principais desafios e pontos de atenção no caso deste projeto ter continuidade futuramente.

2. ESTADO DA ARTE

Define-se um robô segundo a International Organization for Standardization (ISO) como um manipulador multifuncional reprogramável projetado para mover objetos ou realizar diferentes tarefas por meio de movimentos variáreis programáveis. Como uma subclasse de robôs existem os robôs sociáveis, que executam as mesmas tarefas de outros robôs, porém em um contexto de interação social. A definição de interações sociáveis é imaterial e muito abrangente, indo desde tarefas simples de apoio como passar ferramentas, até tarefas mais complexas e expressivas de comunicação e colaboração.

Além disso o aspecto da interação social apresenta uma ligação direta com humanos e a sociedade, o que consequentemente define uma série de valores, regras e padrões sociais relacionados a cada cultura que nós humanos já estamos acostumados, porém isto gera uma dependência cultural para os robôs sociáveis. Assim um robô sociável deve interagir de acordo com as regras sociais implementadas nele, e essas regras são definidas pela sociedade na qual ele é criado.

Robôs sociais começaram a ser pesquisados e desenvolvidos no início dos anos 90 por pesquisadores de inteligência artificial e robótica que desenvolveram robôs que explicitamente se relacionavam com humanos em um nível social. Projetar um robô sociável é particularmente desafiador, uma vez que o robô necessita interpretar corretamente as ações de pessoas e reponde-las de maneira apropriada. Ainda não existe um robô completamente sociável. Cientistas buscam avançar no campo do desenvolvimento de androides e na implementação de habilidades humanas de comunicação para robôs, como entonações e expressões faciais, para desenvolver a tecnologia necessária para o aprimoramento dos robôs sociáveis.

Assim podemos definir Robôs sociáveis como robôs autônomos que interagem e se comunicam com humanos ou outros agentes físicos autônomos seguindo comportamentos sociais e regras definidas em sua programação. Robôs sociáveis como os demais robôs necessitam de um corpo físico para serem considerados robôs. Assim para algo ser considerado um robô sociável é necessário que possua um “corpo” onde tenham motores e sensores que executem alguma função para o robô.

Essa subclasse de robôs foi idealizada para trabalhar em conjunto com humanos em ambientes de trabalho colaborativos, mas está caminhando para atender necessidades pessoais dos humanos em áreas como o cuidado da casa, assistência em hospitais e casas de repouso e até na educação. Como o objetivo deste projeto é desenvolver um módulo para um robô sociável voltado à automação residencial encerra-se aqui a discussão sobre os robôs sociáveis uma vez que este é um aspecto secundário ao desenvolvimento deste projeto. Deste ponto em diante o foco será em sistemas que implementam o diálogo por voz humano-máquina visando a automação residencial ou a execução de outras tarefas, sendo o principal ponto de atenção as suas arquiteturas e funcionalidades.

2.1. ASSISTENTES PESSOAIS PARA AUTOMAÇÃO RESIDENCIAL PRESENTES NO MERCADO

A utilização de assistentes pessoais está se tornando cada vez mais comum, com a popularização de smartphones, que os possuem, como Siri para o IOS e Bixby da Samsung. Com isso, é possível a realização de atividades apenas com comandos verbais, como ligações e pesquisas.

Outra vertente dos assistentes pessoais que está sendo desenvolvido pelas maiores empresas no ramo de eletrônicos é a automação residencial, em que o controle de aparelhos smart, como televisores, home theater e lâmpadas, pode ser feito através de um assistente pessoal doméstico. O Google Home do Google, Echo da Amazon e Homepod da Apple são os maiores representantes do mercado atualmente.

Além de serem capaz de executar comandos por voz em língua inglesa padrões, eles também são capazes de controlar equipamentos smart desenvolvidos por outras empresas que tenham sido conectados previamente. Eles Também possibilitam a criação de comandos personalizados por meio de uma conta de desenvolvedor e da programação da habilidade em seus sites ou aplicativos desenvolvidos para tal tarefa e disponibilizados pelas respectivas empresas.

Ao comparar esses três dispositivos disponíveis no mercado podemos identificar diferenças iniciais ligadas primeiramente aos seus softwares. O Google Home utiliza o software de assistente pessoal desenvolvido pela Google que permite acesso às ferramentas de busca e à base de dados da empresa. Além disso possibilita de forma simples a conexão com smartphones que utilizem o sistema android e o usuário ainda pode vincular à sua conta Google caso já possua. Já o Amazon Echo utiliza a assistente Alexa pertencente à Amazon, o que lhe garante compatibilidade com os serviços de vendas da Amazon, permitindo ao usuário realizar compras pelo dispositivo. Por fim o Homepod da Apple utiliza a assistente Siri, a mesma dos smartphones da Apple. Este último possui menos funcionalidades que os outros dois assistentes por dois principais motivos, o primeiro é por ter sido introduzido no mercado posteriormente aos demais, e o segundo deve-se a Apple ser bastante restrita em permitir a integração de produtos e softwares de terceiros com sua assistente.

Em relação à compatibilidade com os dispositivos Smarts a maioria dos fabricantes produz equipamentos compatíveis com os três dispositivos. Os dispositivos que têm compatibilidade com apenas um dos assistentes são os equipamentos smarts fabricados pelas próprias empresas dos assistentes (Amazon, Apple e Google) e desejam manter o monopólio do equipamento. Todos os assistentes residenciais cobrem as funções básicas esperadas desse tipo de equipamento, mas o Amazon Echo destaca-se em quantidade de funções oferecidas ao usuário em comparação aos seus concorrentes. Já o Google Home permite, quando se tem mais de um aparelho dele instalado em sua residência, transmitir mensagens de voz para outros ambientes da casa que tenham um Google Home instalado e funcionando na mesma rede smart da residência. Por outro lado, o Homepod da Apple fica atrás de ambos concorrentes em relação às ações disponíveis ao usuário, não oferecendo nenhuma outra utilidade de relevância que seus concorrentes não tenham.

O Google Home leva vantagem sobre os assistentes da Apple e Amazon no reconhecimento de voz devido a todos os anos que a Google aplicou em desenvolvimento de reconhecimento de voz e ferramentas de pesquisa e coleta de dados, o que permite que seu assistente responda a diversas perguntas feitas pelos usuários. Enquanto o Homepod fica atrás, uma vez que a Apple ainda está trabalhando para alcançar seus concorrentes e é esperado que a Siri evolua com o tempo. Mesmo assim, todos oferecem um bom reconhecimento de voz, que é o ponto vital desses dispositivos. Na maioria das vezes eles entenderão o que o usuário diz e qual a tarefa que eles precisam realizar.

Ao comparar o valor dos três dispositivos, observamos que o Homepod atualmente é disponibilizado em apenas uma versão de 299 dólares ou então no Mercado Livre por mais de R$3000,00. Porém existem rumores que a Apple está desenvolvendo um novo modelo para seu assistente e repensando o framework utilizado para ser compatível com um maior número de dispositivos.

O Google Home apresenta 4 versões, o Google Home mini, a versão tradicional, a versão max e o mais novo modelo o Google Home Hub, que apresenta tela touchscreen. Com as diferentes versões a Google consegue atender diferentes faixas de preços e expandir a sua base de clientes. A versão mais barata é o Google home Mini encontrado por pouco mais de R$ 150,00 diversas lojas brasileiras, e sua versão mais cara é a versão Max que é vendida apenas no exterior por 400 dólares, porém é possível encontrar essa versão anunciada no Mercado Livre a preços maiores que R$ 3000.

O Amazon Echo é o dispositivo que possui maior número de modelos diferentes, atualmente está disponível em 5 modelos vendidos oficialmente no Brasil, Echo Dot, Echo Dot com relógio, Echo, Echo Show5 e Echo Show8. Além dessas versões atuais diversas outras já foram lançadas, mas são mais facilmente encontradas no exterior. O Echo Dot é o modelo mais barato vendido oficialmente no Brasil custa R$ 249,00, e o modelo mais caro é o Echo Show8 que custa R$899,00, os demais modelos custam preços intermediários a esses.

A proposta deste projeto é criar um sistema similar aos citados anteriormente no quesito da automação residencial, porém que possibilite a criação de rotinas, que

executam mais de uma ação com um simples comando pré-configurado pelo usuário, através de comandos verbais, permitindo que pelo diálogo com o sistema o usuário crie a rotina que desejar. Assim o objetivo é desenvolver um método mais simples de criar rotinas que não necessita de aplicativo ou site.

2.2. ORGANIZAÇÃO DE SISTEMAS DE DIÁLOGO HUMANO-MÁQUINA POR VOZ

Um sistema de diálogo por voz orientado à execução de tarefas é arquitetado de forma modular e unidirecional, composto pelos módulos básicos de um captador de áudio, reconhecedor de voz, um semantizador, um gerenciador de diálogo, um gerador de linguagem natural e um módulo de síntese de voz.

Seu funcionamento é dado da seguinte forma, o captador de áudio basicamente digitaliza o sinal analógico sonoro e o envia ao módulo de reconhecimento de voz, esse por sua vez processa o sinal e gera uma hipótese de reconhecimento no formato de texto que é encaminhada ao semantizador. O semantizador recebe o texto que foi gerado com base no sinal e o analisa utilizando técnicas de processamento de linguagem natural para mapear a fala para algo com significado no diálogo estabelecido gerando uma representação semântica correspondente.

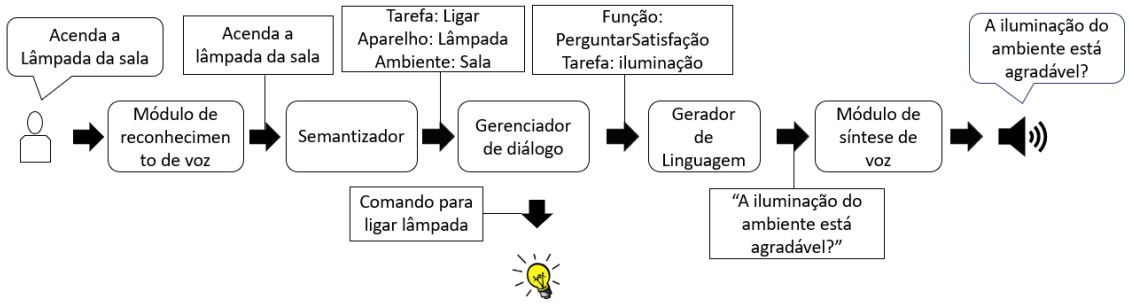

Figura 1: Arquitetura modular básica de um sistema de gerenciamento de diálogo para automação residencial. Autoria própria.

Essa representação semântica é encaminhada ao gerenciador de diálogo que baseado no contexto e dados recebidos define qual é a próxima ação do sistema, esse módulo, é o componente principal do sistema pois controla as atividades de todos os demais componentes e o fluxo do diálogo, sendo ele o responsável por implementar todas as funcionalidades do robô.

O módulo de geração de linguagem natural transforma o comando recebido pelo gerenciador de diálogo em uma intenção de fala no formato de texto retirada de uma base de possíveis respostas do sistema. então finalmente o texto gerado é encaminhado ao módulo de síntese de voz para ser transformado em sinal sonoro para usuário que recebe para o comando que foi dado.

Da mesma forma que o diálogo, o processamento de dados é por sua natureza incremental, fazendo com que a maioria dos sistemas de diálogo sejam tratadas de maneira semelhante [4]. Um sistema incremental é aquele que inicia o processamento

com uma quantidade mínima da entrada padrão, não necessitando da entrada completa para tal, já o sistema não incremental necessita de toda a entrada para iniciar seu processamento. Os sistemas de diálogo por voz podem ser incrementais ou não, porém os sistemas incrementais possuem um tempo de resposta menor pois iniciam o processamento do comando dado antes do término do recebimento.

Como visto um diálogo é estabelecido por seus participantes tomando turnos de fala, e que usualmente depende do contexto, assim podemos enxergar o diálogo como um processo de trocas de informação onde a iniciativa pode trocar entre usuário e o robô, e quem toma a iniciativa conduz o rumo da conversa. Assim podemos classificar os sistemas de diálogo por voz em três tipos diferentes, com base em quem toma a iniciativa e conduz a evolução do diálogo, são eles:

- Iniciativa do sistema: Sistema guia o diálogo em cada passo.
- Iniciativa do usuário: Usuário controla o rumo do diálogo e sistema responde a qualquer comando que o usuário disser.
- Iniciativa mista: O sistema possui maior controle sobre o rumo do diálogo do que o usuário, entretanto ainda é possível ao usuário mudar o rumo do diálogo.

Os sistemas comerciais em sua maioria são projetados como sistemas de iniciativa do sistema, pois o fato do sistema controlar o diálogo torna mais fácil contextualizar os comandos de voz recebidos e isso possibilita um melhor reconhecimento de voz do sistema. Mas quanto mais complexa for a tarefa maior será o número de estados definidos necessários para completá-la e isso os torna mais lentos na execução de tarefas mais complexas.

Já os sistemas de iniciativa mista apresentam um controle de turno de fala mais complexo possibilitando a eles serem mais flexíveis quanto aos comandos do usuário, e dadas suas características eles possibilitam um diálogo entre humano e máquina mais próximo de um diálogo entre humanos.

Devida a importância do módulo de gerenciamento de diálogo, os sistemas de diálogo por voz orientados a execução de tarefas podem ser classificados de três diferentes formas de acordo com o método utilizado para o controle da conversa, são eles:

- Sistemas de estado finito: O usuário é guiado através de uma sequência pré-definida de estados com transições bem claras que definem diversos possíveis caminhos para o diálogo. Esse tipo de sistema é menos flexível à entrada do usuário, porém possui melhor reconhecimento de voz pois os estados podem indicar claramente qual é o contexto do comando recebido.
- Sistemas baseados em frames: O sistema pede informações ao usuário conforme necessário para se completar um template que tem como objetivo cumprir uma tarefa. Nesse caso o fluxo de diálogo não é pré-determinado e depende dos dados fornecidos pelo usuário e das informações que o sistema já possui.
- Sistema de inteligência artificial: Permitem uma comunicação complexa entre o usuário, o sistema e a aplicação, costumam ser de iniciativas mistas e podem ser diversos. Atualmente são utilizadas técnicas de Machine Learning no desenvolvimento desse tipo de sistemas.

Essas são as possíveis arquiteturas e classificações que compõem o estado da arte quando se fala em sistemas de diálogo por voz orientando a execução de uma tarefa. A seguir são apresentados dois frameworks open-source que facilitam o trabalho com esse tipo de sistemas, o primeiro é o RavenClaw, framework de gerenciamento de diálogos, e o segundo é o Olympus, framework completo para o desenvolvimento do sistema de diálogo por voz que utiliza o RavenClaw como seu gerenciador de diálogos.

2.3. FRAMEWORKS EXISTENTES

Diversos frameworks para o gerenciamento de diálogos já foram desenvolvidos. Serão apresentados a seguir frameworks já utilizados na estruturação de diversos gerenciadores de diálogos e Chatbots que propõem formas de estruturar o gerenciador que permitem uma maior modularidade e reusabilidade dos módulos desenvolvidos.

2.3.1. RAVENCLAW

Como gerenciador de diálogo, o RavenClaw fornece a interface falada entre o robô e o usuário. Para que isso seja feito, ele se baseia em planos hierárquicos, que podem ser obtidos por diversos caminhos, se assemelhando assim à estrutura de uma árvore.

O principal diferencial do framework em questão é a capacidade de isolar aspectos da lógica do controle do diálogo voltada à execução da tarefa, das habilidades de conversação, fazendo assim com que seja possível que os desenvolvedores criem uma lógica de diálogo enquanto o Ravenclaw trata os erros, temporização e turnos de fala de forma independente.

O diálogo é estruturado por um plano hierárquico de interação que consiste em uma árvore de agentes de diálogo. Cada agente controla uma subparte da interação que compõem o diálogo. O plano hierárquico estrutura caminhos do diálogo para a execução de uma tarefa, mas não pré-define uma ordem fixa de execução, a ordem de execução pode ser alterada conforme ocorre a interação usuário-máquina.

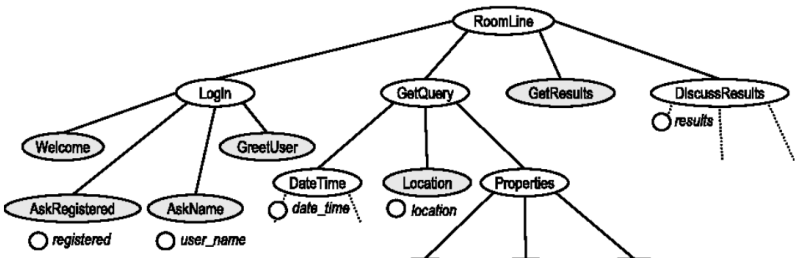

Figura 2: Exemplo de árvore hierárquica de agentes. Retirado da referência [3].

Os agentes de diálogo são de dois tipos. Agentes de diálogo fundamentais cujas as funções principais são: informar, requisitar, esperar informação e executar, e agentes

de instância de diálogo cuja função é controlara execução dos subagentes e encapsular a estrutura lógica e temporal da tarefa de diálogo. Assim agentes fundamentais ocupam posições terminais na árvore e implementam ações de diálogo ou tarefas, enquanto agentes de instância de diálogo ocupam posições não terminais da árvore e controlam o rumo do diálogo.

As informações manipuladas pelo sistema durante o diálogo são encapsuladas em conceitos. Esses podem ser acessados por qualquer agente da árvore, e tem a função de manter o histórico de valores atribuídos e as informações inicias. O algoritmo de execução de uma tarefa do diálogo é centrado por duas estruturas: pilha de diálogo, que captura a estrutura temporal e hierárquica do diálogo, e agenda de expectativas, que controla as entradas esperadas pelo sistema.

A pilha de diálogo é montada com os agentes e subagentes da árvore hierárquica, onde agentes mais no topo da árvore ocupam posições mais baixas da pilha e delimitam o contexto do diálogo, enquanto os agentes no topo da pilha caracterizam o foco do diálogo. Durante os turnos do diálogo novos agentes são movidos para o topo da pilha de diálogo ou removidos da mesma conforme as condições para a execução de sua tarefa são supridas. Esses são eliminados da pilha, a respectiva tarefa é executada e a pilha de diálogo é atualizada. O tratamento dos agentes do topo da pilha pode ser dividido em duas fases de execução e de entradas (quando um agente de requisição é invocado). Assim a pilha de diálogo controla o histórico do diálogo e qual caminho da árvore hierárquica está sendo seguido.

A agenda de expectativas é utilizada nas fases de entrada, onde o sistema fica esperando informações do usuário para atualizar os conceitos e continuar a execução. Os conceitos relacionados aos agentes da pilha são utilizados para montar a agenda de expectativas e facilitar para a máquina a análise semântica da entrada recebida do usuário. Ela é montada lavando em conta a pilha de diálogo, isso define a ordem da lista dos conceitos. Conceitos relacionados à agentes mais no topo da pilha ficam no topo da lista da agenda de expectativas, assim o conceito atualizado é sempre o mais próximo do contexto do diálogo.

Por fim para o tratamento de erros o artigo [3] propõem uma estrutura modular, onde a parte responsável pelo tratamento de erros do gerenciador de diálogos é apenas um outro módulo do gerenciador de diálogo. Essa abordagem aumenta o grau de consistência do sistema e diminui os esforços de desenvolvimento e manutenção do sistema. Os erros inerentes a um diálogo podem ser separados basicamente em dois tipos: não entendimento e mal-entendido.

Erros de não entendimento ocorrem quando o sistema falha em coletar qualquer informação útil do usuário, não é obtida interpretação semântica significativa da entrada do usuário. Erros de mal-entendido acontecem quando o sistema gera uma representação semântica que combina com o contexto, mas não representa a real intenção do usuário. A arquitetura geral do processo de recuperação de erros é composta por um conjunto de estratégias e um processo de decisão que ativam as estratégias.

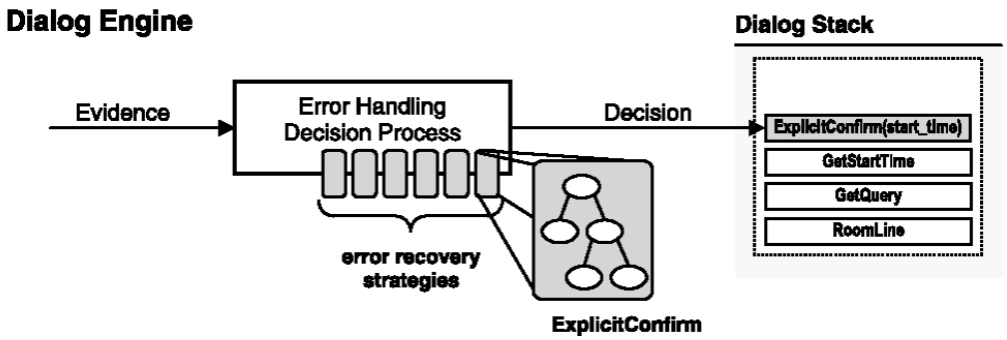

Figura 3: Parte do gerenciador de diálogo responsável pelo tratamento de erros. Retirado da referência [3].

Os conjuntos de estratégias podem ser separados em dois tipos em relação aos possíveis casos de erros, são eles: estratégias para recuperar de erros de não entendimento e estratégias para recuperar de erros de mal-entendido. Assim havendo evidência da ocorrência de um erro, durante a execução de coleta de informações do usuário, o processo de decisão de tratamento de erro é ativado. Constatado o erro e seu tipo a respectiva estratégia de recuperação é ativada e um agente de tratamento de erro é movido para a pilha de diálogo. Assim ele altera o rumo do diálogo para garantir o tratamento do erro e garante que o sistema mantenha informações consistentes.

Assim a organização proposta e o fato de o framework ser open-source traz vantagens como flexibilidade para diversas aplicações, transparência para o desenvolvedor, modularidade e reusabilidade do código desenvolvido, garantindo consistência no funcionamento do sistema.

2.3.2. OLYMPUS

O framework Olympus consiste na utilização de vários módulos para a sua formulação, fazendo com que possua a capacidade de gerar a interface completa de comunicação usuário-máquina, sendo assim, a sua análise é de grande contribuição para que seja criado um sistema similar.

Por ser desenvolvido para propósitos de estudo, o código é open-source (sua codificação pode ser obtidade maneira gratuíta), se tornando assim ideal para estudo.

Para a resolução da comunicação usuário-máquina, o framework possui uma característica principal que é a modularidade, ou seja, é separado em módulos, em que cada módulo é responsável por uma função, como pode ser observado abaixo:

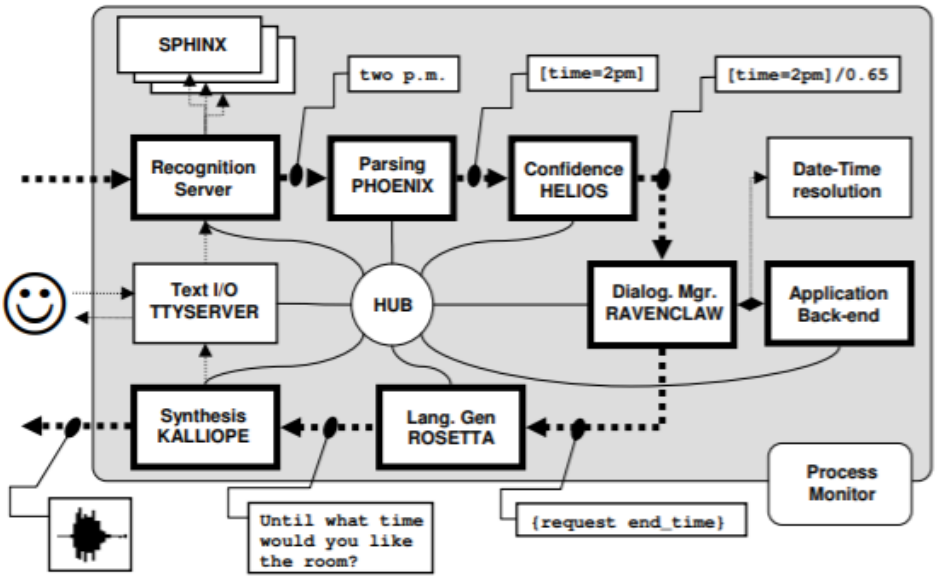

Figura 4: Arquitetura do sistema de diálogo do Olympus. Retirado da referência [5].

Inicialmente, o modulo de Reconhecimenteo de fala (Recognition Server SPHINX) capta o audio e gera as melhores hipóteses de entendimento do comando dado pelo usuário, que é enviada para o módulo de entendimento de linguagem (language understanding component-PHOENIX) que analisa semanticamente as hipóteses extrai os conceitos mais relevantes.

Esses conceitos são enviados para um módulo de confiança (Confidence annotation module – Helios) que atribui uma nota para cada entrada, refletindo a probabilidade de o entendimento estar correto. Se a informação é confiável, envia a hipótese para o gerenciador de diálogo (dialog manager – Ravenclaw), que decide qual a próxima ação a ser tomada.

O módulo de geração de linguagem (language generation module – Rosetta) produz a resposta a ser dada através das concatenações entre diferentes módulos pré-definidos que finalmente é enviada ao módulo de síntese de voz (speech synthesis module – Kalliope), que gera a resposta em formato de áudio que será dada ao usuário.

3. TECNOLOGIAS UTILIZADAS

Para a criação do trabalho, serão utilizadas tecnologias de utilização gratuita disponíveis na Web. Algumas tecnologias têm sua gratuidade sujeita a um determinado consumo máximo, porém isso não foi fator limitante ao desenvolvimento do projeto. Caso o objetivo desse projeto fosse a sua aplicação comercial seria necessário repensar quais tecnologias utilizar. Nas subseções a seguir são apresentadas as tecnologias utilizadas para o desenvolvimento do projeto.

3.1. JADE

O framework JADE (Java Agent Development Framework) possibilita o desenvolvimento de aplicações multiagentes, em que cada agente é responsável por uma função e esses agentes podem se comunicar entre si de modo a realizar uma ou mais tarefas.

Uma característica do JADE é que pode possuir vários hosts, que se comunicam por Java RMI (remote method invocation). Cada host é considerado uma plataforma, que possui um conjunto de containers, em que cada contêiner possui um conjunto de agentes.

Anteriormente a inicialização do projeto, são criados modelos de agentes, que possuem características em comum e é definido como os diferentes tipos de agente se comunicam entre si. Ao iniciar o projeto, são criados alguns agentes padrão, responsáveis por facilitar a utilização do usuário, sendo esses o AMS (Agent management system), ACC (Agent communication channel) e DF (Directory facilitator).

O AMS é um agente gerenciador do sistema, que possibilita a criação, exclusão e suspensão de outros tipos de agentes, que foram desenvolvidos na criação do projeto. Para a comunicação entre os agentes, é utilizado o ACC, que é responsável pela comunicação entre agentes dentro e fora da plataforma, enquanto o DF permite que cada agente encontre o outro de acordo com a tarefa que cada um desempenha.

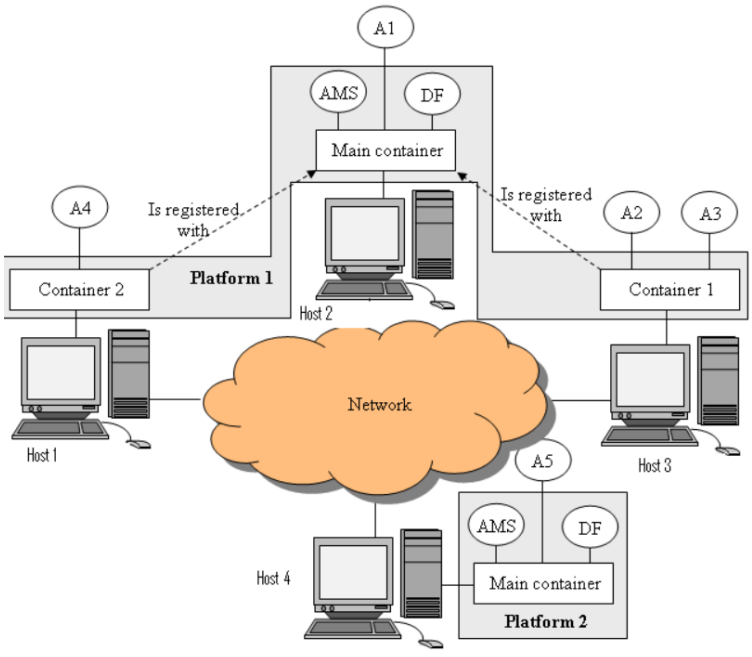

Figura 5: Exemplo de estrutura de um projeto construído através do JADE. Retirado da referência [14].

As características descritas acima permitem com que, através do JADE, seja possível a criação de um projeto modular, facilitando assim a detecção de erros e a mudança de características de cada agente.

3.2. AIML

O AIML (Artificial Intelligence Markup Language) é uma linguagem focada na automatização de respostas, tornando-se muito utilizada para a criação de chatbots. A sua estrutura é baseada em categorias, em que são declaradas respostas (templates) para determinadas entradas (patterns).

A linguagem possui diferentes maneiras de reduzir o número de casos de uso, como a declaração de sinônimos e de reduções simbólicas, que consiste na redução de uma interação do usuário para uma interação já tratada.

Para que a experiência usuário-máquina seja mais natural, o AIML também conta com a possibilidade de gravação de informações dadas pelo usuário, a possibilidade de fornecer diferentes respostas para a mesma pergunta e a utilização de respostas condicionais, ou seja, que dependem do contexto da conversa.

3.3. APIS TEXT-TO-SPEECH E SPEECH-TO-TEXT

Serão utilizadas APIs de Text-To-Speech e Speech-To-Text do Google de sua plataforma de serviços Google Cloud que, entre outros serviços, possui bibliotecas que possibilitam o reconhecimento cognitivo. Essas APIs tem um limite de uso gratuito de 4 milhões de caracteres mensais para conversão de texto para voz e 60 minutos mensais para a conversão de voz para texto, portanto não gerarão custos ao projeto. O funcionamento delas é baseado em Web Services, onde os serviços necessários são acessados por meios de requests aos servidores do Google que respondem o resultado do serviço prestado.

Assim a API de Text-To-Speech funciona pelo envio de um request ao servidor com as variáveis que contém o texto desejado que deseja ser reproduzido e a respectiva língua que deve ser utilizada. Como resposta é obtido um arquivo de som do tipo mp3 referente ao texto enviado. Para a API de Speech-To-Text o request ao servidor possui um arquivo do tipo wav e a língua gravada no áudio, retornando uma variável de texto contendo a transcrição do áudio gravado.

3.4. RDF, RDFS, SPARQL E JENA

Resources Description Framework (RDF) é uma estrutura para a representação de informações na Web. RDF permite realizar afirmações sobre qual quer coisa, tanto concretas quanto abstratas. Essas coisas são denominadas recursos e podem ser uma pessoa, uma empresa, um objeto, um sentimento, uma cor, etc.

As afirmações consistem em três elementos e são chamadas de triplas. Uma tripla possui a seguinte estrutura <sujeito> <predicado> <objeto>. A afirmação RDF

representa uma relação entre dois recursos, sujeito e objeto, e o predicado representa a natureza desta relação formada de forma direcional, do sujeito para o objeto, e é chamado em RDF de propriedade. A seguir é exibido um exemplo geral de estrutura RDF.

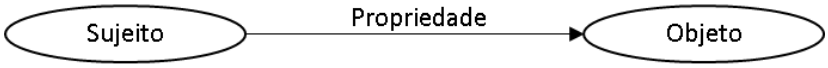

Figura 6: Estrutura geral de uma tripla RDF.

O modelo RDF permite fazer afirmações sobre recursos, mas não permine fazer suposições sobre a semântica dos recursos identificados. Não existe construtor RDF que especifique se um recurso é uma propriedade, ou seja um livro, uma pessoa, etc. Na prática, RDF é normalmente usado junto a vocabulários que forneçam informações semânticas sobre os recursos. Assim o Resources Descripition Framework Schema (RDFS) é um vocabulário que estende RDF e adiciona uma camada onde é possível especificar algumas características que introduzem semântica a dados em RDF.

RDFS possuem construtores que permitem especificar que determinados recursos indicam propriedades de outros recursos, ou então que determinado recurso pertence a uma determinada classe. Assim os recursos podem ser classificados utilizando os construtores RDFS. O vocabulário RDFS permite uma grande variedade de classificações e subclassificações como classes, subclasses, propriedades e subpropriedades e até restrições podem ser definidas por meio de especificações de domínios e contradomínios.

Os dados representados na Web Semântica utilizando o modelo de dados conceitual de RDF, junto com as extensões de RDFS podem estar armazenados em um banco de dados de triplas, um banco de dados relacional com um esquema de mapeamento para RDF. A linguagem SPARQL (SPARQL Protocol and Query Language) é uma linguagem de consulta da Web Semântica e possui uma sintaxe adequada a consultas de dados relacionas representados como um conjunto de triplas RDF.

A linguagem SPARQL admite uma série de filtros e operadores que possibilitam consultas de maior complexidade ao conjunto de triplas armazenada, e diversos bancos de dados de triplas oferecem pontos de acesso via Web que aceitam o protocolo SPARQL e sua linguagem de consulta.

A API Java para RDF Jena foi desenvolvida para aplicações de Web Semântica. Ela implementa interfaces que facilitam o trabalho com modelos RDF em linguagem Java. A API Jena também fornece interface para realizar consultas aos modelos RDF com o uso da linguagem SPARQL. Além disso a API ainda é compatível com a linguagem SPARQL 1.1 que permite remover e adicionar triplas ao modelo RDF, permitindo assim que seja possível atualizar o modelo RDF durante a execução do código.

3.5. OWL E ONTOLOGIAS

Ontology Web Language (OWL) é uma recomendação da W3C de linguagem para definir e instanciar ontologias na Web. O OWL estende RDF e RDFS e oferece um amplo conjunto de tipos de restrições ao conjunto de triplas definidas. São também oferecidos diversos construtores que permitem, entre outras coisas, a construção de classes complexas a partir da definição de classe, e o encadeamento de propriedades. As funcionalidades adicionais da linguagem OWL facilitam a especificação de Ontologias.

Ontologias são especificações explicitas de conceituações que organizam informações semânticas como a base de conhecimento do domínio específico da aplicação [14], ou seja, um modelo de dados que representa um conjunto de conceitos dentro de um domínio. O conjunto de conceitos é composto basicamente por quatro tipos de elementos principais, são eles classes, indivíduos, propriedades e atributos.

Classes são grupos de indivíduos que podem ser utilizadas para escalar a definição das relações entre os indivíduos. Propriedades delimitam o relacionamento entre diferentes classes e indivíduos, e podem ser utilizadas para estabelecer conexões entre eles. Indivíduos são os objetos do domínio e os atributos são características pertencentes aos indivíduos.

O uso de ontologias no gerenciamento de diálogos permite uma interação mais flexível entre o sistema e o usuário. Isso depende apenas da abrangência utilizada na especificação da ontologia do domínio do diálogo, quanto mais abrangente a ontologia maior a flexibilidade possível no diálogo. Esse tipo de estruturação de dados pode ser utilizado no contexto de automação residencial para descrever a relação entre os dispositivos e serviços, e gerenciar as configurações de software.

No artigo [15] os autores propõem uma arquitetura para automação residencial baseada no uso de ontologias. É proposto um sistema com quatro principais bases ontológicas, são elas: ontologia de dispositivos, ontologia de funções, ontologia de preferências e ontologia de ambientes.

A arquitetura proposta por eles torna o sistema mais flexível ao realizar uma mesma função. Por exemplo o sistema pode compreender funções como “ilumine mais a sala”, pois na base ontológica de funções está instanciada a função “iluminar” que se relaciona com dispositivos como lâmpadas e persianas. Assim ao executar a tarefa requisitada o sistema pode decidir por abrir as persianas, acender a lâmpada da sala ou fazer ambos.

3.6. PROTÉGÉ

O Protégé é uma plataforma open-source de ferramentas para a construção de modelos e aplicações baseadas em conhecimentos com o uso de ontologias. A definição das ontologias inerentes aos objetos do diálogo voltado à automação residencial será

instanciada nessa plataforma. O uso de ontologias possibilitará uma maior riqueza e detalhe no diálogo entre o usuário e o sistema.

A plataforma oferece uma maneira mais simples e gráfica de instanciar e editar as classes, propriedades, relações e indivíduos presentes na ontologia a ser definida. As ontologias instanciadas no Protégé podem ser exportadas tanto para o formato de RDF (Resource Descreption Framework) ou OWL, possibilitando posteriormente o acesso a elas por meio de querys.

4. ANÁLISE DE REQUISITOS DE PROJETO

A pesquisa realizada pela Adobe Digital Insights [17] indica um aumento de 103% das vendas de assistentes de voz ao comparar o último trimestre de 2016 com o de 2017 nos Estados Unidos, demonstrando assim o crescente interesse nessas tecnologias. Atualmente, não há grandes empresas focadas no desenvolvimento de robôs sociáveis compatíveis com a língua portuguesa. Com o foco no público brasileiro, o desenvolvimento do projeto será realizado atendendo os requisitos apresentados a seguir e atenderá a comandos em língua portuguesa.

4.1. CONTROLE DE DISPOSITIVOS

O robô deve ser capaz de atender a pedidos básicos de controle de dispositivos smart por comandos de voz do usuário, ou seja, ele deve ser capaz de compreender quando o usuário solicita acionar algum dispositivo e gerar uma resposta adequada para o comando recebido. A resposta adequada pode ser uma pergunta ao usuário para complementar as informações recebidas, a informação que não foi possível realizar a tarefa, o acionamento do dispositivo ou a simulação de seu acionamento.

O acionamento de um dispositivo smart real pelo sistema é secundário a esse projeto, pois o foco deste trabalho é desenvolver uma maneira para que os usuários consigam criar rotinas para o sistema por uma série de comandos de voz. Assim o foco deste projeto é o controle e entendimento do diálogo entre o usuário e o sistema.

4.2. EXTRAÇÃO DE INFORMAÇÕES

O comando dado pelo usuário pode ser considerado insuficiente pelo gerenciador de diálogo. Nesse caso, deve ser iniciado uma rotina para que os dados sejam todos fornecidos.

As informações relevantes para realizar uma automação residencial simplificada, que atenda apenas alguns comandos básicos são:

- O dispositivo alvo que sofrerá a ação;
- qual ação deve ser feita no dispositivo; e
- a localização do dispositivo

Para a obtenção dessas informações, o gerenciador de diálogo deve interagir com o usuário até possua todas as informações necessárias para realizar a tarefa requisitada pelo usuário. Por exemplo, caso o comando dado pelo usuário seja 'ligue o televisor', o robô deve interagir com o usuário para obter qual televisor o usuário se refere, caso exista mais de um televisor desligado cadastrado no sistema.

4.3. CRIAÇÃO DE ROTINAS

O módulo construído para esse trabalho deve ser capaz de utilizar e criar rotinas, ou seja, a interação com mais de um dispositivo através de apenas um comando. Para que a implementação dessa funcionalidade seja feita de maneira simples, a interação verbal com o usuário para tal tarefa não será flexível de modo a facilitar a extração das informações necessárias do usuário.

Decidimos por implementar a criação de rotinas através do histórico de tarefas executadas com sucesso pelo usuário. Assim após seguidas interações de sucesso com usuário realizando diversas tarefas o sistema será capaz, por meio de novas interações, de agrupar as tarefas sob o nome de uma rotina e futuramente executá-la quando for requisitada. Para isso o usuário necessita apenas informar quantas das tarefas executadas com sucesso anteriormente correspondem à nova rotina estanciada por ele.

Usuário: Ligue a televisão da sala
Robô: Televisão da sala ligada.
Usuário: Ligue o home theater.
Robô: Home theater ligado.
Usuário: Feche as cortinas.
Robô: Deseja que eu feche as cortinas da sala?
Usuário: Sim.
Robô: Cortinas da sala fechadas.
Usuário: Criar nova rotina.
Robô: Como gostaria de chamá-la?
Usuário: Assistir a um filme
Robô: O que envolve essa rotina?
Usuário: As últimas três tarefas executadas.
Robô: Rotina 'Assistir a um filme' criada com sucesso

4.4. ARMAZENAMENTO DE INFORMAÇÕES

Para que a experiência do usuário seja mais completa, o projeto também envolve o armazenamento de informações sobre a residência, como qual lâmpada corresponde

a determinado cômodo ou o nome do cômodo. Da mesma forma que a criação de rotinas, essas informações devem ser salvas em um ou mais banco de dados acessíveis pelo sistema. As bases de dados que guardam as informações específicas dos dispositivos, da casa e os conhecimentos necessários para o diálogo compõem a base de conhecimento do sistema.

Para o sistema ser capaz de aprender novas rotinas e gerenciar de forma eficiente os dispositivos ele deve ser capaz de acessar e alterar sua base de conhecimentos.

4.5. DEMAIS FUNCIONALIDADES NÃO IMPLEMENTADAS

A criação de um módulo de robô sociável para automação residencial completo engloba um grande número de funcionalidades, sendo assim, não serão todas abordadas na realização deste projeto.

A seguir serão descritas as possíveis funções que não serão implementadas visando a facilitação de uma possível continuação e melhoria do projeto.

4.5.1. PROGRAMAÇÃO DE AÇÕES

A programação de ações consiste em programar o robô para que realize uma ação no horário, período ou dia desejado. Um exemplo de utilização seria a abertura das cortinas as 6 da manhã.

4.5.2. MODIFICAÇÃO NO BANCO DE DADOS

A criação e cadastro, no banco de dados, de novos dispositivos e locais não pode ser realizada de forma verbal, ou seja, não foi desenvolvido um comando através do qual o usuário é capaz de adicionar uma lâmpada na varanda, ou que adicione um cômodo com nome determinado pelo usuário, como “quarto do João”.

4.5.3. ADQUIRIR INFORMAÇÕES DO SISTEMA

Não será desenvolvido a funcionalidade do sistema em que o usuário é capaz de solicitar informações ao sistema. Essa capacidade, para a automação residencial, pode ser utilizada para adquirir informações a respeito dos diversos dispositivos, como qual dispositivo está ligado, listagem de cômodos, ou qual a temperatura do termostato por exemplo.

4.5.4. APAGAR, EDITAR E LISTAR ROTINAS CRIADAS

Não serão desenvolvidas as funcionalidades do sistema para que o usuário seja capaz de apagar e editar as rotinas criadas. Também não será foco deste projeto desenvolver uma forma para o usuário descobrir quais as rotinas que já foram criadas e estão salvas no sistema, assim o sistema não terá forma de informar ao usuário quais rotinas já foram criadas. O objetivo do projeto é desenvolver a criação de rotinas por comandos de voz sendo a habilidade de editar, apagar ou listar as rotinas algo secundário ao escopo deste projeto. Por outro lado, se o intuito deste projeto fosse sua aplicação comercial, essas funcionalidades seriam fundamentais em um bom produto.

4.5.5. INCLUSÃO E EXCLUSÃO DE DISPOSITIVOS

Outra funcionalidade não implementada neste projeto, porém de grande utilidade em uma aplicação comercial, é a inclusão e exclusão de dispositivos no sistema. Não será abordado uma forma prática para os usuários incluírem e excluírem dispositivos ao sistema. Os dispositivos serão tratados como aspectos estáticos do ambiente para facilitar o desenvolvimento do projeto.

5. DETALHAMENTO DA SOLUÇÃO EMPREGADA

Da seção anterior é possível identificar os casos de uso que o sistema deve atender para que os requisitos traçados sejam atingidos. A Imagem a seguir apresenta o diagrama de casos de uso do sistema. A partir da determinação dos requisitos e dos casos de uso, e do resultado da pesquisa bibliográfica realizada foi determinada uma estrutura geral para o sistema.

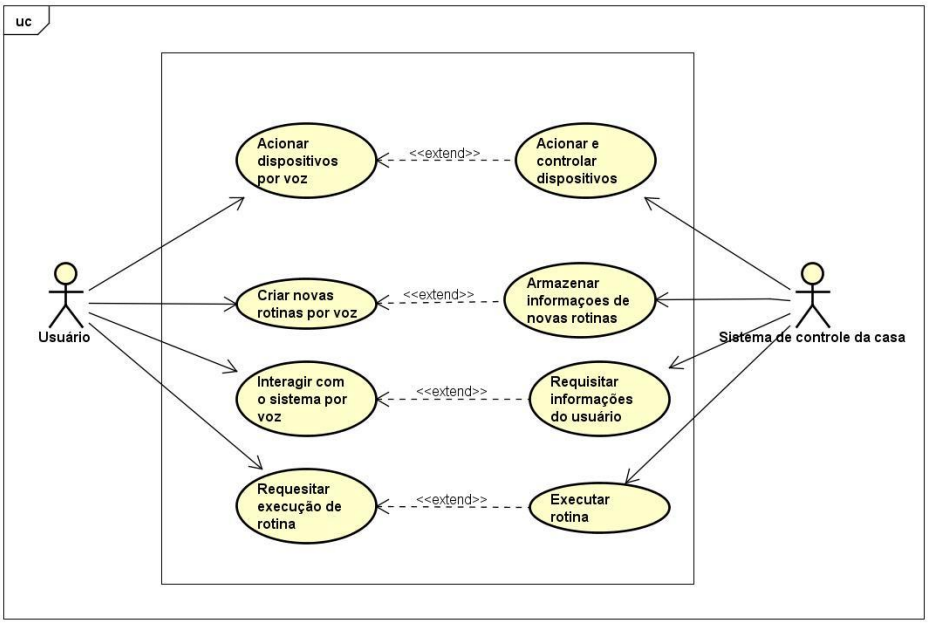

Figura 7: Diagrama de casos de uso do sistema.

A estrutura básica utilizada no desenvolvimento do projeto segue padrão semelhante ao descrito no Estado da Arte (seção 2), seguindo uma arquitetura modular, onde cada módulo pode ser trabalhado de forma independente. Foram adicionados módulos extras de forma a segmentar melhor as funcionalidades e tarefas executadas pelo sistema. Os diferentes módulos representados podem ser agentes do framework Jade, caso seja necessário que sua execução seja contínua, ou apenas classes Java acessadas por suas funções, no caso em que o módulo não necessite persistir por toda a execução. A figura a seguir representa a arquitetura escolhida para o sistema melhor detalhada.

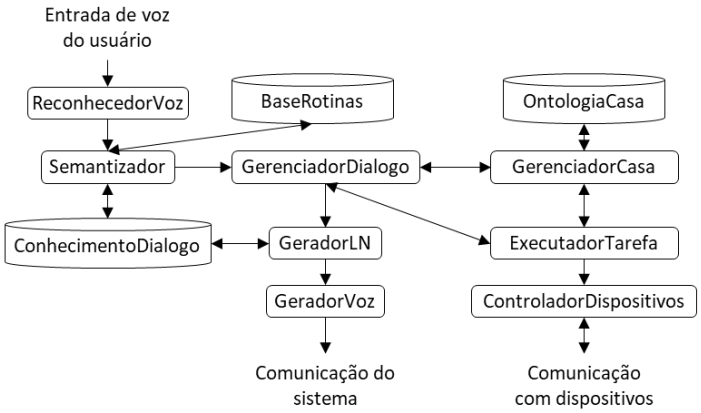

Figura 8: Estrutura proposta para o sistema.

Nas subseções seguintes serão detalhados os módulos que necessitam maior atenção por apresentarem uma maior complexidade em seu funcionamento. De forma complementar ao esquema da estrutura do sistema foi elaborado o diagrama de componentes que apresentam maior detalhamento da estrutura proposta.

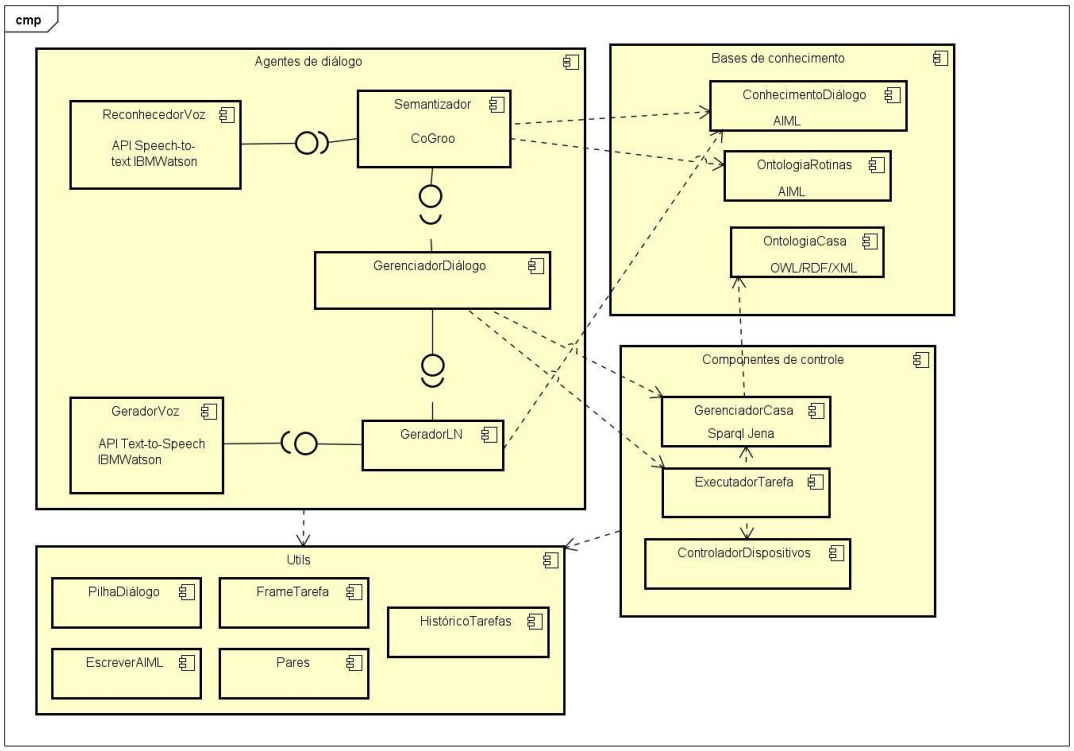

Figura 9: Diagrama de componentes do sistema.

Os componentes estão divididos em quatro grandes grupos são eles:

- Agentes de Diálogo – são agentes do framework Jade de comportamento cíclico responsáveis pela implantação das funcionalidades de controle do diálogo humano-máquina.
- Base de conhecimento – estruturas lógicas que carregam definições a respeito de construções dialogais e do ambiente do sistema.
- Componentes de controle – componentes responsáveis pela implantação de funcionalidades de controle da casa e das rotinas, outras funcionalidades não diretamente ligadas ao diálogo.
- Utils – estruturas de dados de auxilio no tratamento de dados internos do sistema.

Os agentes de diálogo se comunicam através da interface disponibilizada pelo framework Jade. As consultas aos arquivos AIML são realizadas através da interface, desenvolvida pela Alice A.I. Foundation, program-ab que é uma API que simplifica a implementação de Chatbots em linguagem Java com o uso da linguagem AIML. A estrutura AIML do projeto também é utilizada para guardar as rotinas criadas pelo usuário e seu funcionamento será explicado adiante no texto. Por fim a consulta à ontologia da casa é realizada por meio da interface implementada pela API Jena utilizando a linguagem SPARQL. De maneira simplificada o diagrama a seguir ilustra a sequência de ações para o atendimento do caso de uso mais simples de acionar um dispositivo.

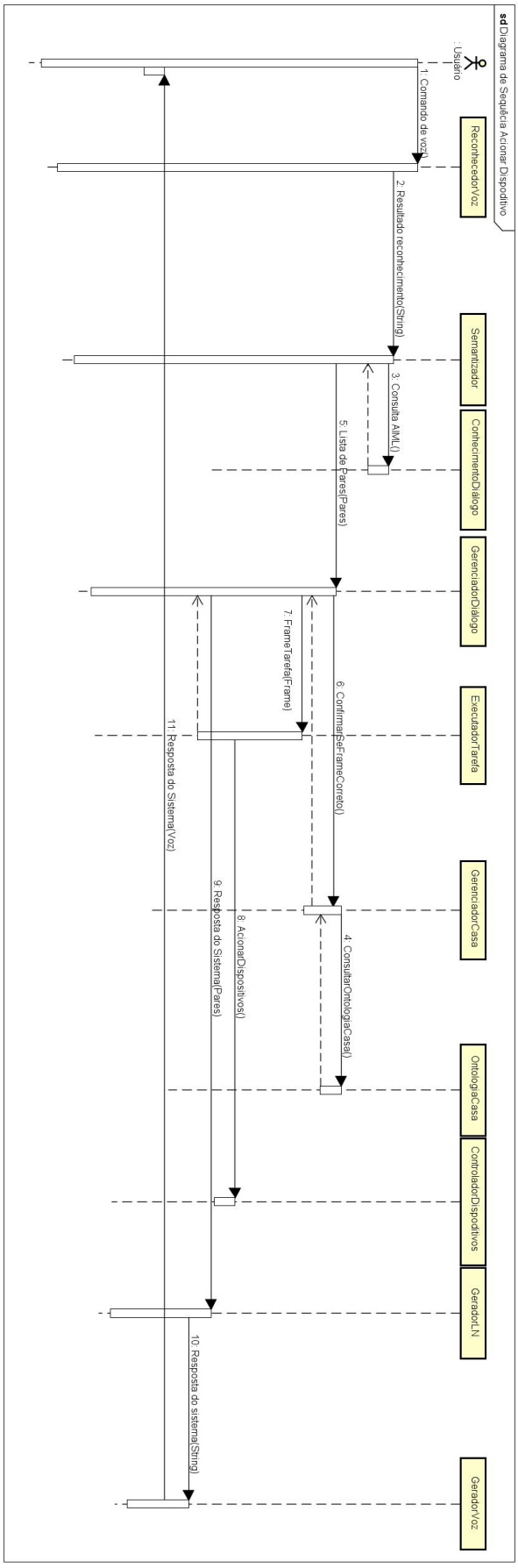

Figura 9: Diagrama de sequência do sistema para acionar um dispositivo.

5.1. BASE DE CONHECIMENTO DO SISTEMA

A base de conhecimentos do sistema compreende três diferentes estruturas. A primeira denominada ConhecimentoDiálogo, consiste em um Chatbot desenvolvido utilizando tecnologia AIML, cuja a função é sistematizar todo o conhecimento de diálogo desenvolvido e adquirido pelo sistema. Suas principais funções são auxiliar no compreendimento de diferentes formas de comunicação que o usuário possa utilizar e guardar as diferentes possíveis formas do sistema se comunicar com o usuário.

A outra estrutura, OntologiaCasa, consiste em uma Ontologia para a casa definida no software Protégé, e compreende toda a base de conhecimento sobre o ambiente em que o sistema está inserido. Nessa base ontológica são definidos quais cômodos a casa possui, quais dispositivos cada cômodo possui e as características, as propriedades, funcionalidade e valores associados a cada dispositivo.

Por fim a BaseRotinas é a parte do sistema onde as informações para a execução das tarefas correspondentes de cada rotina são armazenadas. Cada rotina possui um nome único e n Frames completos. O Frame é uma estrutura criada que possui os argumentos necessário para a execução de uma tarefa como a ação, dispositivo e local, essa estrutura é explica em maior detalhe mais adiante no texto. As rotinas criadas pelos usuários são salvas em um arquivo no formato AIML onde o nome da rotina define o *pattern* e o número de frames completos junto com os respectivos frames que a compõem estão definidos no *template* da estrutura AIML. Desta forma o Semantizador é o responsável por identificar quando está sendo solicitada a execução de uma rotina, e nestes casos recuperar e enviar ao gerenciador os frames que caracterizam a rotina em questão para que seja executada.

5.2. SEMANTIZADOR

Inicialmente, o reconhecedor de voz da Google utilizado no projeto transcreve o comando de voz dado pelo usuário e informa a confiabilidade do texto. Esses dados então são enviados através de uma String pelo JADE para o módulo Semantizador.

O Semantizador é responsável por analisar se a confiabilidade é suficiente, analisar sintaticamente a frase, separar os dados que serão utilizados e enviá-los ao Gerenciador de Diálogo. Para que todas as etapas sejam realizadas, primeiramente é feita a análise da frase através do Cogroo, que identifica o significado de cada palavra e como elas se relacionam, como pode ser observado na figura a seguir, onde é utilizada como exemplo a frase “Ligue a lâmpada do quarto”:

Erros gramaticais

Nenhum erro gramatical encontrado.

Análise morfológica

Nº.	Elemento	Lemas	Classe	Flexão	Sintagma	Função
1	Ligue	ligar	verbo finitivo	presente, #1/3S#, subjuntivo	– verbal	– predicado
2	a	o	artigo	feminino, singular	┌ nominal	┌ objeto direto
3	lâmpada	lâmpada	substantivo	feminino, singular	└	|
4	de	de	preposição		– preposicional	|
5	o	o	artigo	masculino, singular	┌ nominal	|
6	quarto	quarto	substantivo	masculino, singular	└	└

Árvore sintática

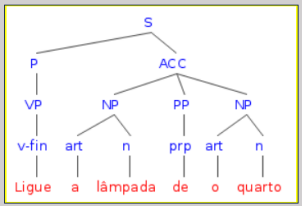

Figura 10: Exemplo de funcionamento do Cogroo.

A análise da confiabilidade oferecida pelo reconhecedor de voz é feita após a análise sintática, pois o Cogroo possui a capacidade de classificar sintaticamente todos os elementos de uma frase, mesmo com possíveis erros, como pode ser observado no exemplo a seguir, em que o entendimento do Reconhecedor de Voz foi incorreto, transcrevendo a frase para “Líder a lâmpada do quarto”.

Erros gramaticais

Nº.	Regra	Mensagem curta	Substituir por
1	space:EXTRA_BETWEEN_WORDS	Excesso de espaços entre as palavras.	

Análise morfológica

Nº.	Elemento	Lemas	Classe	Flexão	Sintagma	Função
1	lider		verbo particípio	masculino, singular	– verbal	– predicado
2	a	a	preposição		– preposicional	┌ objeto preposicional
3	lampada		substantivo	masculino, singular	– nominal	|
4	a	a	preposição		– adverbial	|
5	de	de	preposição		– preposicional	|
6	o	o	artigo	masculino, singular	┌ nominal	|
7	quarto	quarto	substantivo	masculino, singular	└	└

Árvore sintática

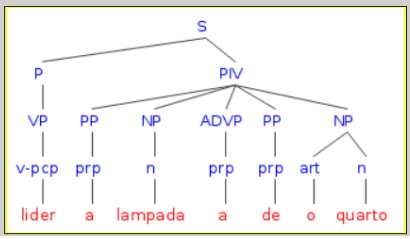

Figura 11: Exemplo de funcionamento do Cogroo com erro no entendimento.

Retiradas as palavras que possuem baixa confiabilidade, as demais são separadas de acordo com sua função semântica. Para que a análise do Gerenciador de Diálogo seja facilitada, as palavras de interesse passam por um processo de lematização e uma análise em que o Chatbot, definido pela base de ConhecimentoDiálogo, é utilizado para identificar sinônimos, e assim padronizar as diferentes formas de enunciar um mesmo comando. Por exemplo, o comando “Acenda a luz do quarto” é modificado para “Ligar a lâmpada do quarto”.

Com a frase padronizada e as funções semânticas das palavras identificadas, o Semantizador gera os pares dialogais, que são formados pela junção da Intenção, que consiste em uma String especificando qual informação será enviada, e do Argumento, ou seja, o conteúdo semântico. Alguns exemplos de pares que são gerados podem ser observados na tabela a seguir.

Tabela 1: Exemplos de atos dialogais utilizados na comunicação com o Gerenciador de Diálogo.

Intenção	Argumento	Exemplo de fala do usuário	Argumento do exemplo
Informar a função	"Criação de rotinas" ou "gerenciar dispositivos"	"Ligue a lâmpada do quarto."	"Gerenciamento de dispositivos
Informar a tarefa	Ação		"Ligar"
Informar o dispositivo	Dispositivos		"Lâmpada"
Informar o local	Local		"Quarto"
Confirmação	"Sim" ou "Não"	"Sim"	"Sim"

Para cada ação que o sistema deve realizar é criado uma lista contendo os pares dialogais com as informações obtidas pelo sistema, ou seja, se o comando envolver dois dispositivos, será formado duas listas com pares. Feito isso, as listas são enviadas para o Gerenciador de Diálogo descrito a seguir.

5.3. GERENCIADOR DO DIÁLOGO

O gerenciador de diálogo recebe a lista de pares de atos dialogais e argumentos identificadas pelo Semantizador, e trata par a par para compor o frame e efetuar a execução da tarefa demandada. O gerenciador de diálogo percorre a lista de pares recebidos do Semantizador identificando as informações recebidas de modo a estruturar um frame que irá compor a pilha de diálogo do sistema. O frame dependerá do tipo de tarefa demanda pelo usuário e pode necessitar a completude de diferentes lacunas para a concretização do comando e execução da tarefa.

Tabela 2: Exemplo de frame completo.

Domínio	Tarefa	Ação	Dispositivo	Local
AutomaçãoResidencial	GerenciarDispositivos	Ligar	Lâmpada	Quarto

Vale observar que para a parte do frame na qual é identificado o domínio como “automaçãoResidencial” não será utilizada e não possuí relevância para esse trabalho em específico. Porém a adição desse campo possibilita futuramente a adição de outras funcionalidades para o Robô sociável, ou seja, novos módulos.

O Gerenciador de Diálogo está organizado no formato de uma árvore lógica que percorre cada uma das variáveis que compõem o frame, testando-as e gerando as respostas adequadas para cada situação. As tarefas executadas em cada nó da árvore definem o curso da interação humano-máquina. Neste projeto a árvore foi desenvolvida apenas até o ponto onde é possível a implementar a criação de rotinas por comandos de voz, e, portanto, encontra-se bem simplificada. Mas caso fosse dada continuidade ao projeto bastaria expandir a árvore para as novas funcionalidades do sistema que necessitam ser implementadas.

De forma esquemática a sequência lógica de atividades do Gerenciador de Diálogo pode ser observada na imagem a seguir.

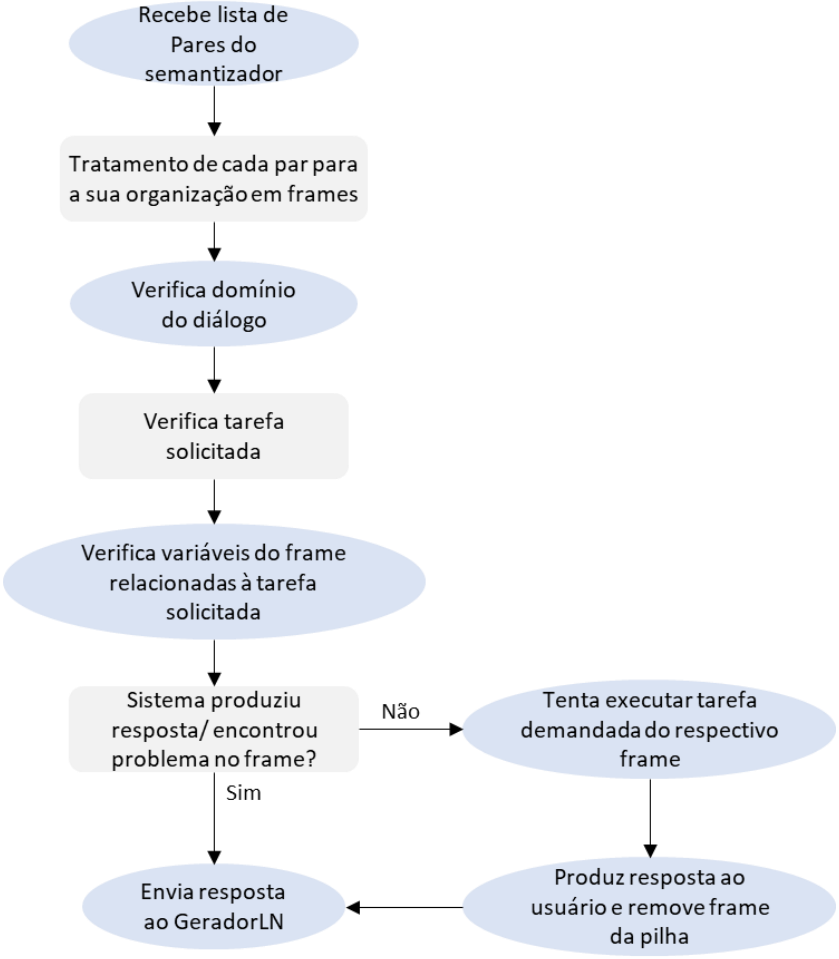

Figura 12: Fluxo lógico das atividades desempenhadas pelo gerenciador de diálogo.

Assim cada variável que compõem o frame influenciará no destino do diálogo. Em cada nó da árvore são realizados testes utilizando a Ontologia da Casa para verificar os argumentos que compõem o frame. Por meio de consultas à ontologia da

casa intermediadas pelo módulo Gerenciador Casa é verificado se o frame em questão é suficiente para a execução da tarefa, ou se mais interações com o usuário são necessárias. Aqui é testada a existência de dispositivos e locais no sistema e se a ação solicitada é possível de ser executada pelo sistema.

Conforme o Gerenciador de Diálogo percorre os nós da árvore o comando solicitado ganha cada vez mais especificidades até o ponto em que pode ser executado. Quando o gerenciador atinge os nós da árvore que permitem a execução do comando, o frame é então enviado ao Executador de Tarefas cuja função é apenas executar a tarefa solicitada pelo frame recebido. O Executador de Tarefas então realiza a tarefa solicitada e retorna ao gerenciador o resultado da tarefa na forma de intenção dialogal que deve ser transmitida ao usuário.

Além disso o gerenciador de diálogo pode necessitar confirmar informações com o usuário a respeito caso ocorram erros de não entendimento ou de mal entendimento, e alguma variável do frame esteja faltando ou não seja entendida pelo sistema. Todas as intenções dialogais geradas pelo gerenciador de diálogo são padronizadas em *Strings* compreendidos pelo Gerador de Linguagem Natural. Essas intenções dialogais são enviadas para o Gerador de Linguagem Natural onde são transformadas em frases estruturadas compreensíveis pelo usuário.

5.4. EXECUTADOR DE TAREFA

Uma vez acionado o Executador de Tarefas irá tentar executar a tarefa representada pelo frame recebido do Gerenciador. Caso não seja possível executar a rotina, é criada pelo sistema uma intenção dialogal que representa a resposta do sistema para o usuário que informa o problema ocorrido e dá continuidade ao diálogo.

Quando a execução da tarefa é possível, o Executador de Tarefas consulta via o gerenciador da casa a ontologia definida para obter o protocolo de comunicação e o endereço do dispositivo a ser acionado no sistema montado. Essas informações junto às informações inerentes à ação solicitada são enviadas ao Controlador de Dispositivos, agente responsável pela interação do sistema com os dispositivos instalados na casa. O Executador de Tarefas recebe a resposta do Controlador de Dispositivos se a tarefa foi ou não executada e transforma isso na String de intenção dialogal que é enviada do Gerenciador de Diálogo para o Gerador de Linguagem Natural.

5.5. GERADOR DE LINGUAGEM NATURAL

O módulo Gerador de Linguagem Natural acessa um banco de dados (ConhecimentoDiálogo) escrito na linguagem AIML em um formato padronizado de modo a permitir que a mensagem estruturada na forma de atos dialogais seja convertida para um String que represente uma frase elaborada em língua portuguesa. Assim de

acordo com o ato dialogal fornecido pelo Gerenciador de Diálogo, o gerador de linguagem natural criar uma frase em língua portuguesa para a comunicação com o usuário. A seguir é apresentado um exemplo do funcionamento do módulo em questão:

Tabela 3: Exemplo de funcionamento do Gerador de Linguagem Natural.

Plano de ação decidido pelo Gerenciador de Diálogo	**Frase formada pelo Gerador de Linguagem Natural**
"Foi ligado " +dispositivo+ " em " + local	“O dispositivo “dispositivo” do local “local” foi acionado com sucesso!

A frase obtida do banco de dados de conhecimento de diálogo é enviada ao módulo Gerador de Voz, para que o String resultante da consulta ao banco de dados seja convertido em voz. A conversão do String em voz é realizada utilizando a API Text-To-Speech do Google Cloud citada anteriormente.

5.6. GERENCIADOR DA CASA

O gerenciador da casa utiliza da linguagem SPARQL e da API Java Jena para realizar consultas e alterações na base de dados da Ontologia da casa. Nesse módulo são implementadas diversas funções que permitem consultar a Ontologia em busca de diferentes informações como a existência de um determinado ambiente na casa, ou de um determinado dispositivo em um ambiente, ou o estado atual de alguma propriedade de um dispositivo, etc. Também são implementadas funções que alteram as propriedades dos dispositivos de modo que conforme as interações prossigam o sistema mantenha atualizado as informações guardadas a respeito da casa e seus dispositivos.

5.7. CRIAÇÃO DE ROTINAS

A criação de rotinas é implantada de forma inflexível de modo a facilitar o processamento dos dados informados pelo usuário. Como exemplificado na seção 4 após o usuário executar as tarefas que compõem a rotina ele deve informar ao sistema primeiramente o desejo de criar uma rotina. O sistema perguntará ao usuário o nome desejado para a rotina e em seguida a quantidade de comandos que a compõem. Com as informações adquiridas do usuário o sistema recupera os últimos *n* comandos do usuário, os agrupa sob o nome da rotina e salva essa estrutura em um arquivo AIML onde são alvas as rotinas criadas. Quando é solicitada a execução de uma rotina os frames que representam os comandos que a compõem são recuperados pela consulta ao arquivo AIML de rotinas e enviados ao gerenciador para sua execução. A seguir é apresentada esquematicamente a lógica implementada para a criação de rotinas e na página seguinte é exibido seu diagrama de sequência.

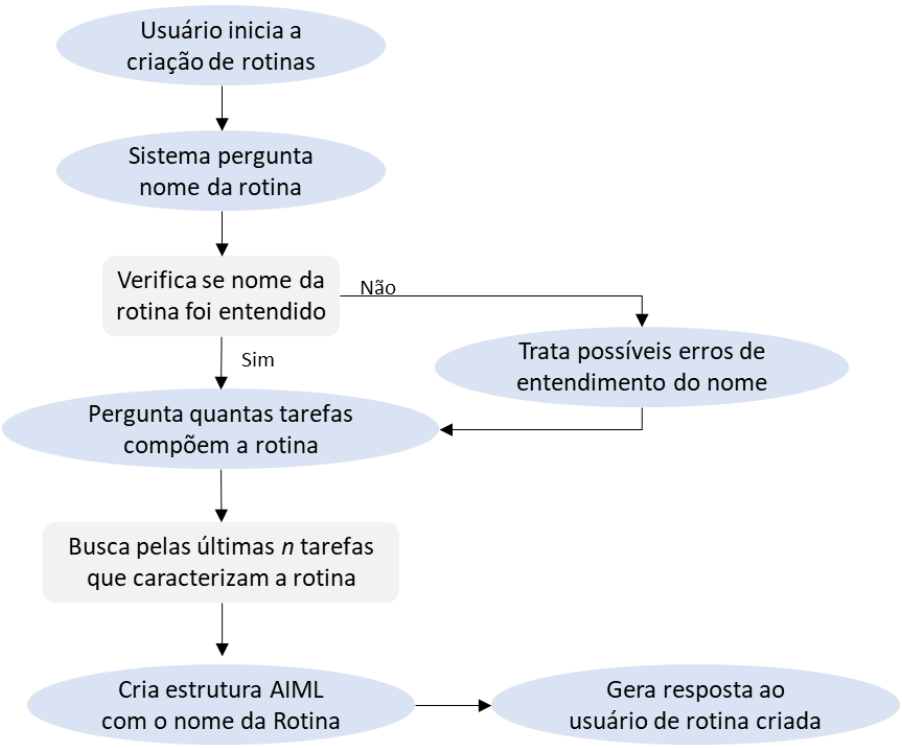

Figura 13: Fluxo lógico das atividades para criação de uma rotina.

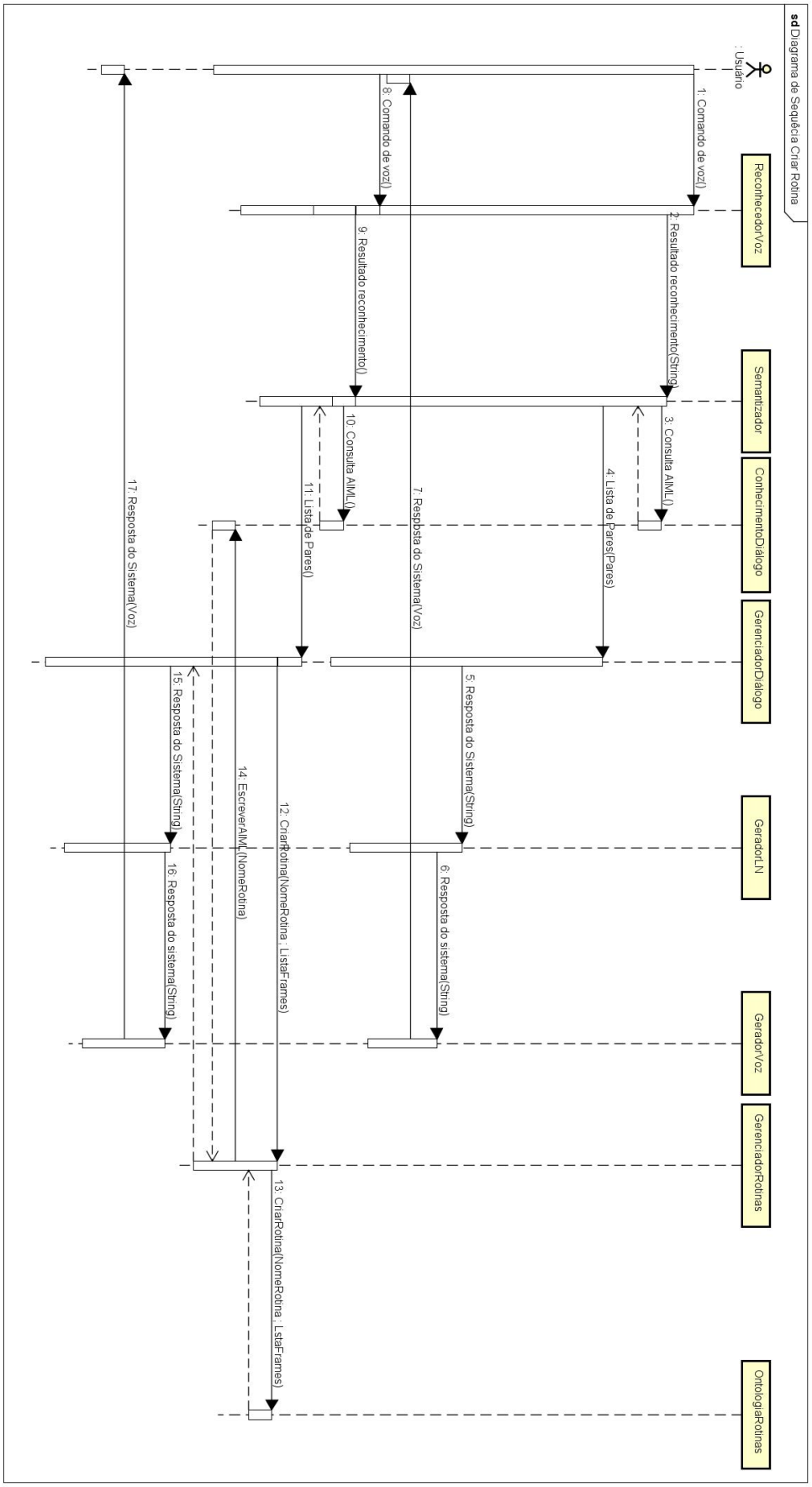

Figura 14: Diagrama de sequência da criação de uma rotina.

6. RESULTADOS

Para aferir os resultados obtidos com o sistema foi criado um ambiente virtual que simula uma casa com dispositivos smarts para serem acionados, visando ilustrar o funcionamento do sistema sem a necessidade do uso de dispositivos smart reais. O ambiente virtual criado é simplificado com poucos ambientes, dispositivos e funcionalidades, e seu propósito é apenas ilustrar e testar a criação de rotinas por comandos de voz.

Assim a casa criada em ambiente virtual possui apenas quatro cômodos, sala, quarto, cozinha e varanda, e três tipos de dispositivos diferentes, lâmpada, televisão e som. A seguir é detalhado com os dispositivos estão espalhados pelos cômodos e quais as funcionalidades que cada um possui.

Cômodos e respectivos dispositivos:

- Quarto: Lâmpada, televisão e som
- Sala: Lâmpada, televisão e som
- Varanda: Lâmpada e som
- Cozinha: Lâmpada

Dispositivos e respectivas funcionalidades:

- Lâmpada: Ligar e desligar.
- Televisão: Ligar, desligar e aumentar ou diminuir o volume.
- Som: Ligar, desligar e aumentar ou diminuir o volume.

Todos os dispositivos encontram-se inicialmente desligados e são representados em uma janela atualizada pelo sistema conforme os estados dos dispositivos exibida a seguir.

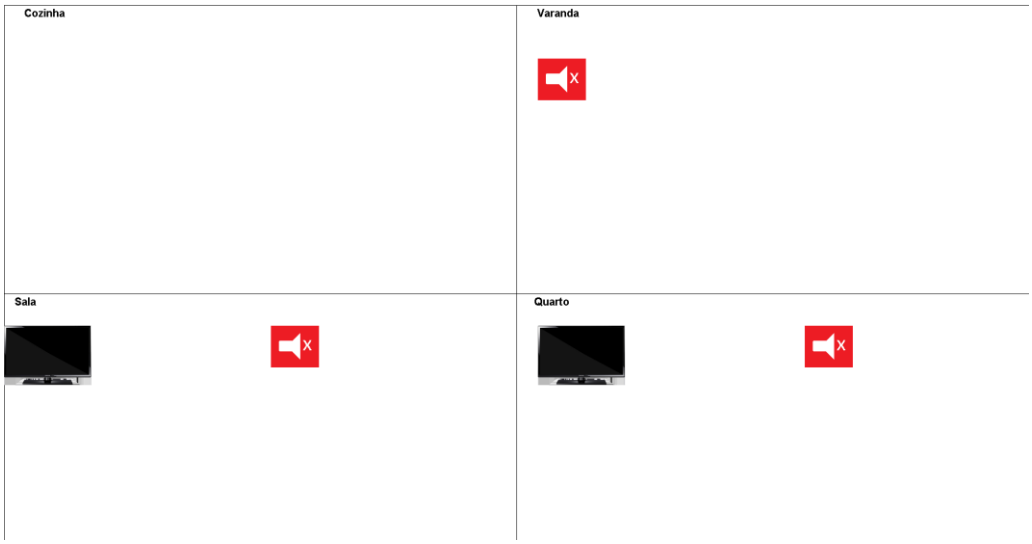

Figura 15: Janela de visualização do ambiente virtual criado em seu estado inicial.

A seguir são exibidas uma série de comandos sequenciais que foram utilizados para testar o módulo desenvolvido e os respectivos resultados apresentados pelo sistema assim como o resultado exibido no ambiente virtual criado.

- Comando simples:
  Usuário: Ligar a luz da sala.
  Resposta módulo: O dispositivo lâmpada do local sala foi acionado com sucesso!

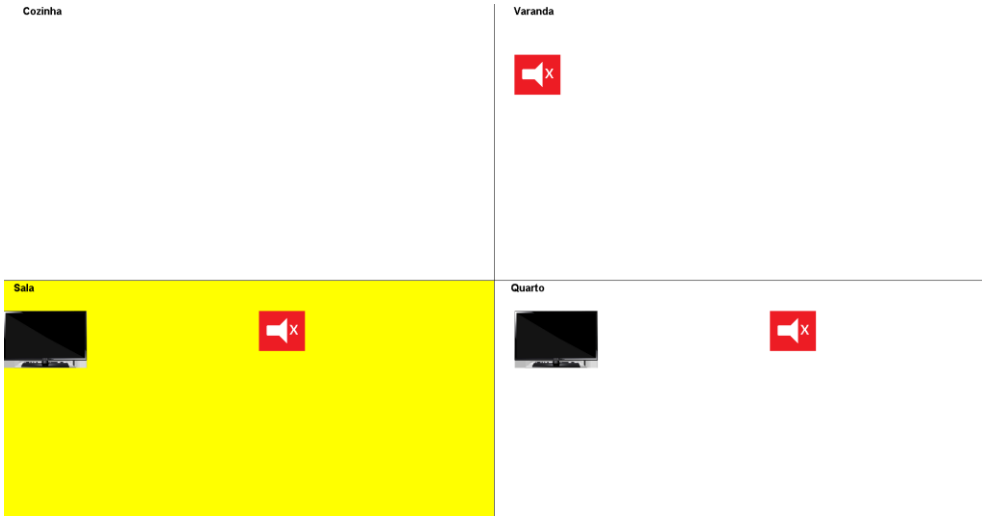

Figura 16: Janela de visualização do ambiente virtual criado após 1º comando.

- Comando composto:
  Usuário: Ligar a televisão do quarto e desligar a luz da sala
  Módulo: O dispositivo televisão do local quarto foi acionado com sucesso!
  Módulo: Dispositivo lâmpada do local sala foi desligado.

Figura 17: Janela de visualização do ambiente virtual criado após o 2º comando.

- Comando composto 2:
  Usuário: Ligar a lâmpada e o som do quarto
  Módulo: O dispositivo lâmpada do local quarto foi acionado com sucesso!
  Módulo: O dispositivo som do local quarto foi acionado com sucesso!

Figura 18: Janela de visualização do ambiente virtual criado após o 3º comando.

- Comando simples sem especificar ação com apenas 1 opção de ação para o dispositivo:
  Usuário: Luz da varanda
  Módulo: Deseja ligar a lâmpada do local varanda?
  Usuário: Sim
  Módulo: O dispositivo lâmpada do local varanda foi acionado com sucesso!

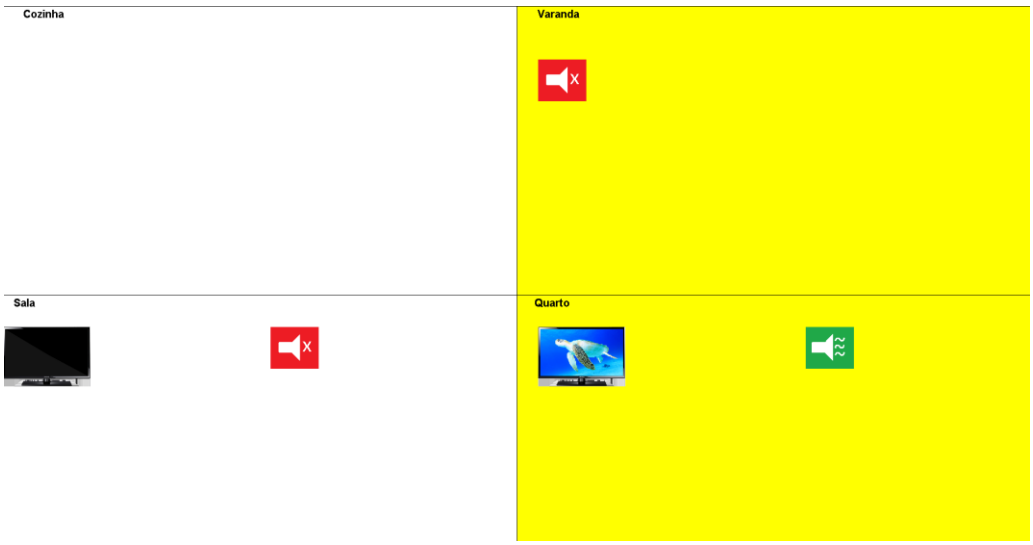

Figura 19: Janela de visualização do ambiente virtual criado após o 4º comando

- Comando simples sem especificar ação com mais de 1 opção ação para o dispositivo:

  Usuário: Som do quarto

  Módulo: A tarefa solicitada para o som do local quarto não foi entendida. Tente um novo comando deixando clara qual tarefa deseja realizar.

  Não há alteração no ambiente virtual.

- Comando simples sem especificar local com mais de 1 opção de dispositivos possíveis (demonstração 2):

  Usuário: Ligar a lâmpada.

  Módulo: O local não foi compreendido. Porém o dispositivo lâmpada foi encontrado nos locais varanda, cozinha, quarto, sala. Tente um novo comando.

  Não há alteração no ambiente virtual.

- Comando simples com funcionalidade não existente:

  Usuário: Aumentar a lâmpada do quarto.

  Módulo: O dispositivo lâmpada não suporta a função solicitada.

  Não há alteração no ambiente virtual.

- Não entendimento do comando:

  Usuário: Executar um comando não existente

  Módulo: (Não há resposta)

  Não há alteração no ambiente virtual.

- Comando simples com 2 ou mais características não compreendidas:

  Usuário: Som

  Módulo: O comando solicitado não foi compreendido. Tente um novo comando.

  Não há alteração no ambiente virtual.

- Comando com ação impossível:

  Usuário: Ligar a televisão do quarto.

  Módulo: Não foi possível acionar o dispositivo televisão do local quarto. Tente outro comando.

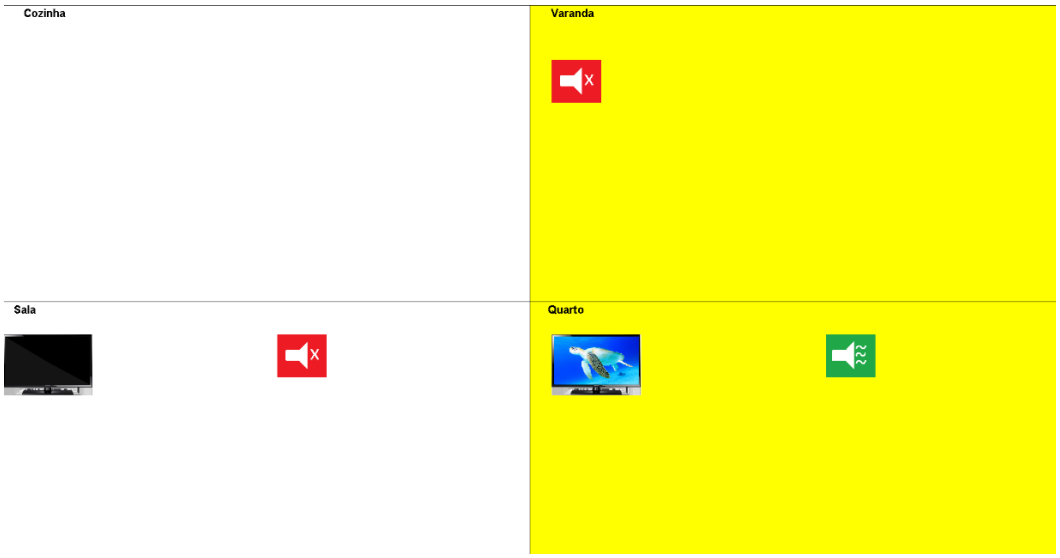

Figura 20: Janela de visualização do ambiente virtual criado após o comando recebido. Não ocorreram alterações devido a erros nos comandos.

6.1. RESULTADOS OBTIDOS PARA A CRIAÇÃO DE ROTINAS

Dado o estado inicial do ambiente virtual exibido a seguir e o exemplo de sequência de comandos para a criação de uma rotina, obteve-se o seguinte resultado:

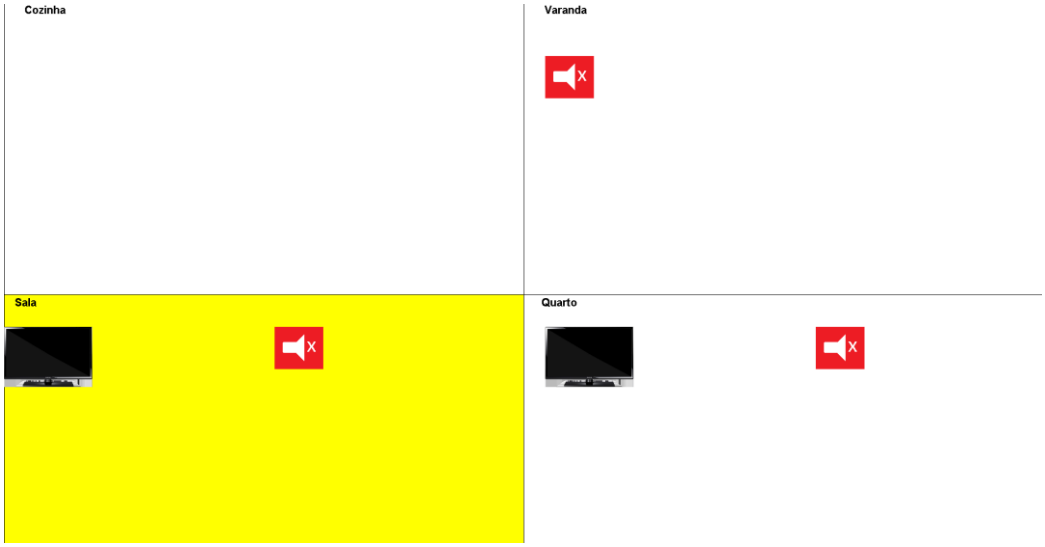

Figura 21: Janela de visualização do ambiente virtual em seu estado inicial.

- Sequência de comandos para criar rotina:
  Usuário: Desligar a luz da sala
  Módulo: Dispositivo lâmpada do local sala foi desligado.
  Usuário: Ligar a televisão e o som da sala.
  Módulo: O dispositivo som do local sala foi acionado com sucesso!

Módulo: O dispositivo televisão do local sala foi acionado com sucesso!
Usuário: Criar uma rotina.
Módulo: Qual o nome da rotina que deseja criar?
Usuário: Assistir um filme.
Módulo: Quantos comandos possui a rotina?
Usuário: 3
Módulo: Rotina com 3 comandos foi criada com sucesso!

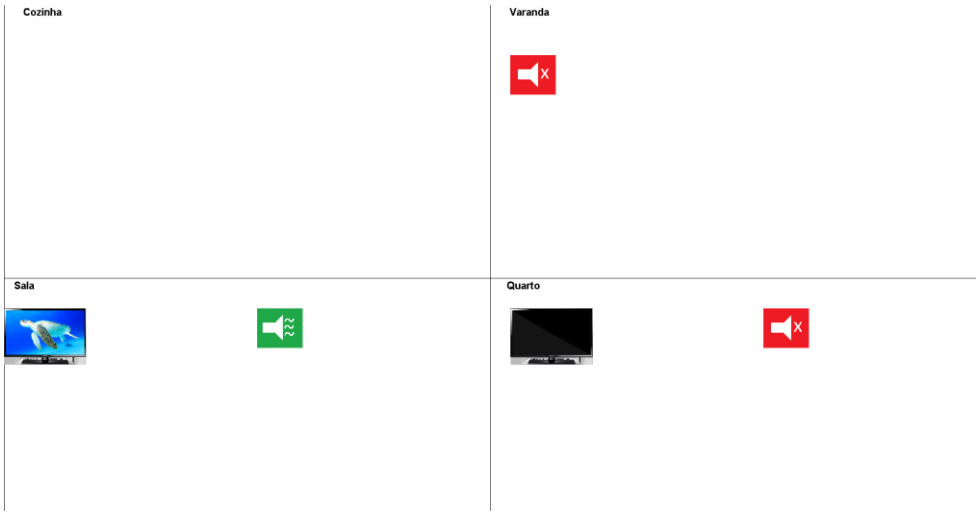

Figura 22: Janela de visualização do ambiente virtual criado, resultado após sequência de comandos.

- Tentativa de criar rotina com erros:

Usuário: Desligar a luz da sala
Módulo: Dispositivo lâmpada do local sala foi desligado.
Usuário: Ligar a televisão e o som da sala.
Módulo: O dispositivo som do local sala foi acionado com sucesso!
Módulo: O dispositivo televisão do local sala foi acionado com sucesso!
Usuário: Criar uma rotina.
Módulo: Qual o nome da rotina que deseja criar?
Usuário: Rotina teste.
Módulo: Quantos comandos possui a rotina?
Usuário: Preço
Módulo: Houve um erro para criar a rotina. Inicie o processo novamente.
Não há alteração no ambiente virtual.

- Exemplo de execução de rotina previamente criada:

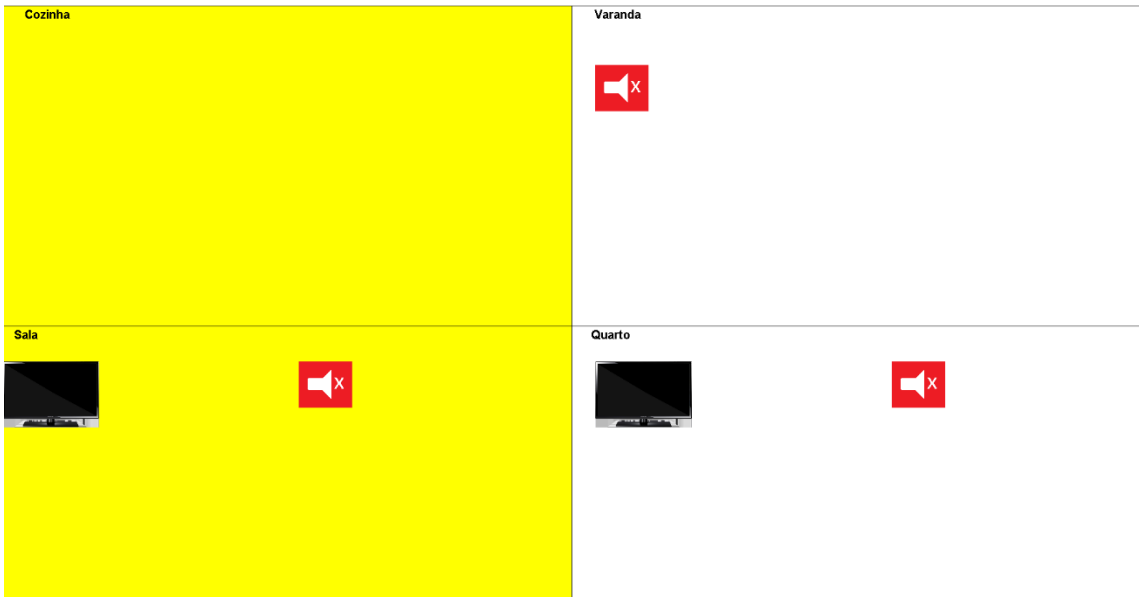

Figura 23: Janela de visualização do ambiente virtual em seu estado inicial.

Usuário: Assistir um filme
Módulo: Dispositivo lampada do local sala foi desligado.
Módulo: O dispositivo televisao do local sala foi acionado com sucesso!
Módulo: O dispositivo som do local sala foi acionado com sucesso!

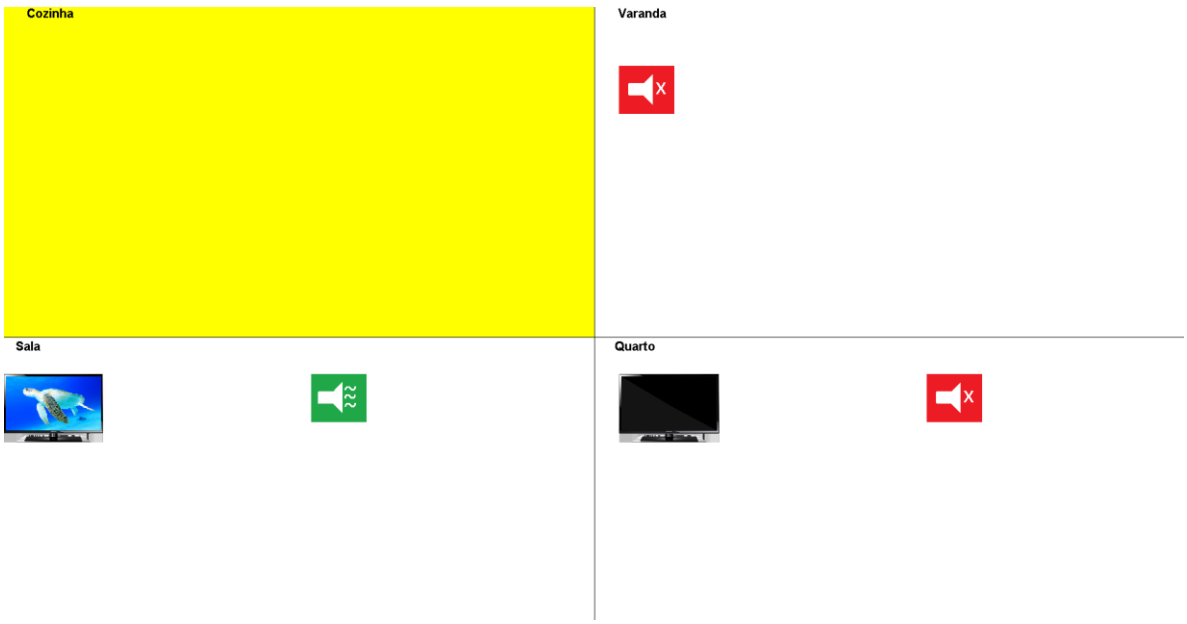

Figura 24: Janela de visualização do ambiente virtual após a execução da rotina.

Como explicado anteriormente não forma implementados neste projeto os casos de apagar rotina, editar rotina, listar rotinas, adicionar dispositivos e remover dispositivos visando baixar a complexidade do sistema a ser desenvolvido e por isso esses casos não foram testados.

7. CONCLUSÕES

A proposta do projeto, que consistia na conceituação e criação do módulo de robô sociável focado na automação residencial foi atingida com sucesso para a utilização em pequena escala. Devido à utilização de API's de terceiros, nos módulos de reconhecimento e geração de voz, para a utilização do sistema em larga escala seria necessário realizar pagamentos aos fornecedores do serviço, o que no contexto da comercialização do módulo terá que ter sua viabilidade analisada.

O projeto possibilita realizar o controle básico de dispositivos cadastrados no sistema por meio de comandos de voz emitidos pelo usuário. Também foi desenvolvida a criação de rotinas com nomes definidos pelo usuário, porém essa funcionalidade foi implementada de forma pouco flexível, onde o usuário necessita seguir um roteiro pré-definido para finalizar a criação com sucesso. Flexibilizar a criação de rotinas por comandos de voz para torná-la mais amigável ao usuário e permitir diferentes formas de criar rotinas é um ponto importante a ser considerado no desenvolvimento futuro do projeto.

A interação com o módulo pode ser futuramente complementada através da implementação de casos de uso que não foram desenvolvidos, como a alteração, listagem e exclusão de rotinas já criadas pelo usuário, da implementação do controle de outros dispositivos, como termômetros e ar condicionados para o controle de temperatura, e também pela implementação de funcionalidades de dispositivos mais complexas como a escolha de uma música, a troca ou escolha de canal da televisão e muitas outras que já são presentes nos assistentes presentes no mercado.

Quanto ao funcionamento do módulo, a integração com novos dispositivos e locais pode ser citada como possível melhoria no projeto, uma vez que hoje o módulo está preparado apenas para o recebimento funcionamento de dispositivos de iluminação, som e televisores localizados na varanda, quarto, cozinha ou sala. O modo que foi construído o módulo foi o suficiente para possibilitar a criação de rotinas por comandos de voz do usuário, porém não possibilita ao usuário flexibilidade para adicionar, editar e remover os dispositivos e os locais presentes no sistema. Assim possibilitar que o usuário altere os dispositivos e os locais do sistema trará um ganho de flexibilidade para o módulo, e, portanto, deve ser considerado em seu desenvolvimento futuro, principalmente se este estiver voltado a sua aplicação comercial.

Assim as principais melhorias a serem feitas no módulo desenvolvido são voltadas a aprimorar a interação usuário-máquina, tornando-a mais flexível. Os principais pontos de atenção para a melhoria do sistema em seu desenvolvimento

futuro devem ser: aumentar a variedade de tarefas que ele é capaz de executar, e desenvolver sua base de conhecimento de diálogo para compreender um maior número de frases do usuário. Desenvolver esses dois aspectos tornará a interação do usuário com o módulo mais flexível e amigável, deixando o diálogo humano-máquina mais fluído, e tornará o módulo mais capacitado para ser futuramente instalado em um robô sociável.

8. REFERÊNCIAS

[1] Cuayáhuitl, H., 2016. SimpleDS: A Simple Deep Reinforcement Learning Dialogue System. arXiv preprint arXiv:1601.04574.

[2] Lee, C.J., Jung, S.K., Kim, K.D., Lee, D.H. and Lee, G.G.B., 2010. Recent approaches to dialog management for spoken dialog systems. Journal of Computing Science and Engineering, 4(1), pp.1-22.

[3] Bohus, D. and Rudnicky, A.I., 2009. The RavenClaw dialog management framework: Architecture and systems. Computer Speech & Language, 23(3), pp.332-361.

[4] Schlangen, D. and Skantze, G., 2009, March. A general, abstract model of incremental dialogue processing. In Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (pp. 710-718). Association for Computational Linguistics.

[5] Bohus, D., Raux, A., Harris, T.K., Eskenazi, M. and Rudnicky, A.I., 2007, April. Olympus: an open-source framework for conversational spoken language interface research. In Proceedings of the workshop on bridging the gap: Academic and industrial research in dialog technologies (pp. 32-39). Association for Computational Linguistics.

[6] McTear, M.F., 2002. Spoken dialogue technology: enabling the conversational user interface. ACM Computing Surveys (CSUR), 34(1), pp.90-169.

[7] Pellom, B.L., Ward, W. and Pradhan, S.S., 2000, October. The CU communicator: an architecture for dialogue systems. In INTERSPEECH (pp. 723-726).

[8] Seneff, S., Hurley, E., Lau, R., Pao, C., Schmid, P. and Zue, V., 1998, November. GALAXY-II: a reference architecture for conversational system development. In Icslp (Vol. 98, pp. 931- 934).

[9] Ward, W. and Issar, S., 1994, March. Recent improvements in the CMU spoken language understanding system. In Proceedings of the workshop on Human Language Technology (pp. 213-216). Association for Computational Linguistics.

[10] Huang, X., Alleva, F., Hwang, M.Y. and Rosenfeld, R., 1993, March. An overview of the SPHINX-II speech recognition system. In Proceedings of the workshop on Human Language Technology (pp. 81-86). Association for Computational Linguistics.

[11] Como criar rotinas para a Alexa. https://www.cnet.com/how-to/control-your-smart-home-with-a-single-command-using-alexa-routines/.

[12] Apple HomePod Vs. Amazon Echo Vs. Google Home: Which Smart Speaker Is Right For You?. https://www.forbes.com/sites/forbes-finds/2018/02/09/apple-homepod-vs-amazon-echo-vs-google-home-which-smart-speaker-is-right-for-you/#72be57df7dec.

[13] Bellifemine, Fabio, Agostino Poggi, and Giovanni Rimassa. "JADE–A FIPA-compliant agent framework." Proceedings of PAAM. Vol. 99. No. 97-108. 1999.

[14] D. GRIMSHAW, Jade administration tutorial, Ryerson University, Toronto, Canadá, março de 2010 <http://jade.tilab.com/documentation/tutorials-guides/jade-administration-tutorial/architecture-overview/>. Disponível em:, acesso em 20/06/2018.

[15] Xu, Jingjing ; Lee, Yann Hang ; Tsai, Wei Tek ; Li, Wu ; Son, Young Sung ; Park, Jun Hee ; Moon, Kyung Duk. / Ontology-based smart home solution and service composition. Proceedings - 2009 International Conference on Embedded Software and Systems, ICESS 2009. 2009. pp. 297-304

[16] García, Guillermo; Amores, Gabriel; Manchon, Pilar; Gómez Marín, Fernando; González Martí, Jesús. 2006, Integrating OWL ontologies with a dialogue manager.

[17] Adobe Digital Insights. Consumer eletronics report. https://files.acrobat.com/a/preview/64813cfc-5680-41f8-9564-dbea4847a952.

[18] Wallace, Richard S., 2003, March.The Elements of AIML Style. ALICE A.I. Foundation, Inc.

[19] PARK, Youngmin; KANG, Sangwoo; SEO, Jungyun. An Efficient Framework for Development of Task-Oriented Dialog Systems in a Smart Home Environment. **Sensors**, v. 18, n. 5, p. 1581, 2018.

[20] JEON, Heesik et al. An Intelligent Dialogue Agent for the IoT Home. In: **AAAI Workshop: Artificial Intelligence Applied to Assistive Technologies and Smart Environments**. 2016.

[21] HUANG, Cheng-Chi; LIU, Alan; ZHOU, Pei-Chuan. Using Ontology Reasoning in Building a Simple and Effective Dialog System for a Smart Home System. In: **Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on**. IEEE, 2015. p. 1508-1513.

[22] Centro de Estudos Sobre Tecnologias WEB – CeWeb, Guia de Web Semântica, http://ceweb.br/guias/web-semantica//capitulo-4/ , visitado em 10/11/2018.

[23] Novo HomePod, controle para jogos, AirPowers… confira as previsões de lançamentos da Apple para 2020/21. .https://macmagazine.uol.com.br/post/2020/04/20/novo-homepod-controle-para-jogos-airpowers-confira-as-previsoes-de-lancamentos-da-apple-para-2020-21/.

[24] Smart Speakers com Alexa. Os dispositivos Echo são caixas de som inteligentes controladas por voz. https://www.amazon.com.br/b?ie=UTF8&node=19877613011#:~:text=Dispositivos%20Echo%20na%20Amazon.com,todos%20eles%20integrados%20com%20Alexa.

[25] Social Robot Article. https://en.wikipedia.org/wiki/Social_robot

[26] World Economic Forum.Top 10 Emerging Technologies 2019, Insight Report. http://www3.weforum.org/docs/WEF_Top_10_Emerging_Technologies_2019_Report.pdf

[27] The 5 Biggest Smart Home Trends In 2020. https://www.forbes.com/sites/bernardmarr/2020/01/13/the-5-biggest-smart-home-trends-in-2020/#6c88ce6a389b

[28] Share, Perry; Pender, John. Preparing for a Robot Future? Social Professions, Social Robotics and the Challenges Ahead. Irish Journal of Applied Social Studies: Vol. 18: Iss. 1, Article 4. doi:10.21427/D7472M disponível em: https://arrow.tudublin.ie/ijass/vol18/iss1/4

[29] Apple, Google, and Amazon are teaming up to develop an open-source smart home standard. https://www.theverge.com/2019/12/18/21027890/apple-google-amazon-smart-home-standard-zigbee-connected-ip-project