UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

MARCELO MAGNO RODRIGUES

Interface de voz para dispositivos domésticos de acessibilidade

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Engenharia da Computação

Orientador: Prof. Dr. Claudio Fernando Resin Geyer
Co-orientador: Desireé Santos

Porto Alegre
2019

“Se tiver o hábito de fazer as coisas com alegria,
raramente encontrará situações difíceis.”

— ROBERT BADEN-POWELL

AGRADECIMENTOS

Gostaria de agradecer aos meus pais Pedro e Oraides, que tornaram esse passo possível, e foram incansáveis em me apoiar nessa longa caminhada. E aos meus irmãos Fabrício e André, que se mantiveram próximos nos momentos menos esperançosos.

Agradeço também ao professor Geyer pela acolhida e e incentivo num momento crucial, e à Desireé pelas cobranças e pela preocupação com o desenvolvimento deste trabalho.

Um agradecimento muito especial ao amigo Mathias Mantelli, por preciosas horas dedicadas, e frequentes palavras de motivação.

RESUMO

Este trabalho apresenta a implementação de uma interface de voz para o controle de um carrossel vertical de prateleiras destinado ao uso por pessoas com desabilidades visuais, físicas ou motoras. Sabidamente pessoas com desabilidades tem maior chance de sofrerem acidentes domésticos, ou serem acometidos por problemas de saúde relacionados a maus hábitos alimentares e sedentarismo. A aplicação de interfaces de voz em dispositivos domésticos pode tornar estes ambientes mais acessíveis, melhorando sua qualidade de vida dessas pessoas. A interface reúne características importantes observados em interfaces de voz definidas em trabalhos semelhantes, e apresenta uma avaliação do impacto destas características na experiência do usuário na interação com a interface.

Palavras-chave: Interface de Voz. Casas Inteligentes. Tecnologia Assistiva. Arquitetura Móvel.

Speech interface for home accessibility devices

ABSTRACT

This work presents the implementation of a voice interface for the control of a vertical shelf carousel, witch in intended for use by people with visual, physical or motor disabilities. Impaired people are known to have a greater chance of having a home accident, or being affected by health problems related to poor eating habits and physical inactivity. Applying voice interfaces to home devices can make these environments more accessible, improving their quality of life. The developed interface puts together important features observed in voice interfaces defined in similar works, and presents an assessment of the impact of these features on the user experience while interacting with the interface.

Keywords: Speech Interface, Smart Homes, Assistive Technology, Mobile Archtecture.

LISTA DE ABREVIATURAS E SIGLAS

MVC *Model View Controller*

MVVM *Model View View-Model*

MVP *Model View Presenter*

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

Acessibilidade descreve em que grau um objeto ou ambiente está disponível para as pessoas que o utilizam, independente de suas necessidades pessoais (IPIÑA; LORIDO; LÓPEZ, 2011). Pessoas com desabilidades carecem de acessibilidade para desenvolver suas tarefas diárias de maneira autônoma e segura. Problemas acarretados por falta de hábitos saudáveis em termos de exercícios e alimentação são mais incidentes em pessoas com necessidades especiais. Isso ocorre pelo cansaço gerado pelo aumento da complexidade na execução das tarefas corriqueiras (RIMMER; ROWLAND, 2008) (SIXSMITH; SIXSMITH, 2008). Há ainda um risco maior de acidentes domésticos e de aparecimento de doenças psicológicas (SIXSMITH; SIXSMITH, 2008). Esses problemas atingem boa parcela da população idosa, o que, dado o aumento da população nessa faixa etária, gera uma demanda crescente por adaptações domésticas. (SIXSMITH; SIXSMITH, 2008) (DEWSBURY; EDGE, 2001).

Auxiliar pessoas com desabilidade a viver de forma mais independente e saudável são objetivos reconhecidos dos estudos da área de *Smart Homes* (PORTET et al., 2013). O nicho de *Smart Homes* que se aprofunda em pessoas com necessidades especiais chama-se *Assistive Homes* (NUSSBAUM, 2006), e como adaptações de acessibilidade são úteis a todos, a indústria emprega essa tecnologia em equipamentos que aumentam a comodidade das pessoas em seus produtos (DEWSBURY; EDGE, 2001). Para atender a diversidade existente na demanda por essa, diferentes formas de interagir com os aparatos tecnológicos de acessibilidade são explorados. Interfaces gráficas de alto contraste, responsivas, e comandadas por voz são desenvolvidas e aprimoradas. Tais abordagens são alternativas ao uso de botões, teclados e painéis convencionais, para pessoas com limitações físicas ou motoras (ROSENFELD; OLSEN; RUDNICKY, 2001).

Dentre as alternativas de interação, o uso da voz é especialmente vantajoso em aplicações de *Smart Homes* e interações ubíquas, pois são menos invasivas em termos de uso e requerem *hardware* simples e portável (ROSENFELD; OLSEN; RUDNICKY, 2001). Do ponto de vista do usuário, interfaces de voz trazem possibilidades não presentes em interfaces gráficas, como não ocupar as mãos do usuário nem sua atenção visual. Com isto a pessoa que interagir com uma interface de voz pode executar outra tarefa simultaneamente. Já para usuários com necessidades especiais, aumenta o ganho em acessibilidade (LINES; HONE, 2006), ao permitir que pessoas com limitações físicas ou motoras possam interagir com aparelhos sem ajuda de terceiros para leituras, manipulação

de botões ou mudanças de posição para acamados (RASHID et al., 2017).

O sucesso de uma interface de voz é definido pela sua capacidade de mediar a interação entre a pessoa e a máquina de maneira completa, e eficiente em relação ao esforço despendido pela pessoa. Esta capacidade está intimamente ligada ao quanto a interface é capaz de atender às expectativas do usuário. Estudos como (KAMM, 1995) e (YANKELOVICH; LEVOW; MARX, 1995a) apontam aspectos importantes das expectativas do usuário a serem considerados durante o desenvolvimento de interfaces de voz. A não observação dessas expectativas pode levar a interações onerosas em relação ao gasto de tempo e a falhas de comunicação, comprometendo a confiança do usuário no sistema (YANKELOVICH; LEVOW; MARX, 1995a).

Durante as pesquisas chegou-se ao trabalho (ARNOLD; KLENNER, 2015), em que os autores projetam uma estante de prateleiras automatizada, desenvolvida para pessoas com desabilidades visuais, físicas ou motoras. O projeto foi desenvolvido visando a acessibilidade, e a principal característica é seu design ergonômico. O móvel foi projetado para que pessoas em cadeiras de rodas tenham acesso itens das prateleiras, independente da sua altura, uma vez que as prateleiras são móveis, e ao toque de botões, a prateleira desejada é posicionada ao alcance confortável do usuário. Outras características, como o emprego de botões grandes, coloridos, e gravados em braile, ou a escolha de materiais seguros contra acidentes, são outras características pensadas para pessoas com algum tipo de necessidade especial. O objetivo do móvel é aumentar o conforto e autonomia dos usuários no ambiente da cozinha. Após conversas com a autora do projeto, notou-se a possibilidade de se empregar uma interface de voz, que além de trazer as vantagens de acessibilidade inerentes a este tipo de interface, poderia estender a aplicação da estante de prateleiras para outros ambientes domésticos além da cozinha.

No cenário de interfaces de voz que implementam boas práticas próprias, sem reunir características vantajosas de trabalhos similares, e da acessibilidade observada no projeto de estante acessível disposto em (ARNOLD; KLENNER, 2015), decidiu-se empregar esforços no desenvolvimento de uma interface de voz. O objetivo central deste trabalho, é reunir as boas práticas observadas em diferentes trabalhos sobre interfaces de voz, numa única interface, no contexto de um dispositivo doméstico de acessibilidade. Para então avaliar em testes com usuários a aceitação destas práticas, e como estas poderias comportar-se sendo aplicadas simultaneamente numa mesma aplicação.

Para chegar em uma interface funcional o suficiente para ser testada por usuário, possibilitar uma avaliação sobre suas características, elencaram-se objetivos específicos

com esta finalidade. Inicialmente a escolha de uma plataforma sobre a qual a aplicação seria implementada, que resultou no uso da plataforma *IPhone* de dispositivos móveis da *Apple* por facilidades de implementação observadas. Propôs-se um conjunto de comandos com o intuito de tornar possível guardar objetos de outros ambientes domésticos, com comandos semanticamente coesos. Propôs-se também uma avaliação sobre padrões de design reconhecidos para se escolher aquele que guiaria a organização dos módulos da aplicação. Esta avaliação resultou na escolha do padrão *Model View View-Model* pela sua afinidade à implementação de aplicações para o sistema operacional *iOS*, dos dispositivos móveis da *Apple*. Propôs-se então uma adaptação deste padrão de design para o uso em interfaces de voz, para lidar com restrições impostas no desenvolvimento deste tipo de interface. Uma arquitetura foi proposta, utilizando o padrão de design adaptado, para coordenar os serviços de reconhecimento de voz, sintetização de voz e reprodução de ícones de áudio, observados como os principais canais de entrada e saída da interface implementada. E o desenho do gerenciador de inventário, que seria o modelo de dados do domínio.

Para se ir do projeto do conjunto de comandos falados à avaliação, propôs-se uma metodologia de desenvolvimento. Esta metodologia foi pensada em quatro etapas. A primeira se definiu por fase inicial de desenvolvimento, visando produzir uma implementação inicial. Nesta implementação inicial objetivou-se a obtenção de uma interface, que, mesmo incompleta, fosse capaz de ser operada por usuários, recebendo e respondendo o conjunto completo de comandos de voz. A segunda etapa consistiu da apresentação da interface a usuários, para que estes interagissem com esta, e sugerissem variações textuais para os comandos que se pretendeu implementar, assim como apontassem dificuldade de interação e possibilidades de melhora. Estas sugestões guiaram a terceira etapa, que consistiu de utilizar as sugestões dos usuário, adicionando as variações textuais sugeridas aos comandos implementados, e fazendo pequenas modificações para melhorar a fluidez no uso da interface. Após a segunda fase de implementação, obteve-se a versão final da aplicação. Esta versão da interface foi então apresentada a um grupo de usuários. Estes testaram a aplicação seguindo um roteiro definido com a finalidade de fazer o usuário passar por todo o conjunto de comandos, e ter contato com as boas práticas escolhidas para serem implementadas na interface. Ao fim dos testes, os usuários foram convidados a preencher um formulário de avaliação, o qual foi analisado para se atingir o objetivo final de identificar o impacto que a implementação do conjunto escolhido de boas práticas teria na usabilidade da interface proposta.

O trabalho está organizado da seguinte forma: o capítulo 2 trata da motivação, e retrata as questões de acessibilidade estudadas, assim como descreve o design contido em (ARNOLD; KLENNER, 2015). O capítulo 3 descreve o objetivo final, e os objetivos específicos elencados para o trabalho. No capitulo 4 trata-se sobre a fundamentação teórica reunida, sobre a área de *Smart Homes* na seção 4.1, sobre interfaces de voz na seção 4.2, sobre a arquitetura na seção 4.3, e sobre outros trabalhos da área na seção 4.4. A aplicação proposta, e sua organização são tratadas no capítulo 5. A metodologia usada para atingir os objetivos de implementação, testes e avaliação é apresentada no 6, e a avaliação dos resultados propriamente dita é abordada no capitulo 7. A conclusão é exibida no capítulo 8, e considerações sobre trabalhos futuros no capítulo 9.

2 MOTIVAÇÃO

Esta seção apresenta as motivações para este trabalho, que abrangem acessibilidade e tecnologia de *smartphones*. A pesquisa se iniciou com a observação da necessidade crescente de ambiente acessíveis e da importância de se viver em um lar adaptado às necessidades pessoais próprias de cada pessoa. Percebeu-se a possibilidade de melhorar a interação de pessoas com seus lares através da adaptação de utensílios domésticos para que estes sejam mais acessíveis.

Ao se buscar por tecnologias de auxílio à acessibilidade, rapidamente se chega na área de automação residencial e, mais especificamente, na área de *Smart Homes*. Entre leituras e conversas com pessoas que desenvolveram trabalhos sobre acessibilidade, chegou-se ao trabalho (ARNOLD; KLENNER, 2015), ao qual se propôs-se uma continuação, através do desenvolvimento de uma interface de voz, que poderia estender suas funcionalidades.

2.1 Acessibilidade e saúde

Um ambiente ou utensílio é considerado mais ou menos acessível de acordo com o quanto este se encontra disponível para todas as pessoas às quais se destina seu uso, independente de suas necessidades pessoais (IPIÑA; LORIDO; LÓPEZ, 2011). Se para interagir com um ambiente uma pessoa depende da presença de outra pessoa para auxiliá-la, isto aponta falta de acessiblidade do objeto ou ambiente em questão. Note que a acessibilidade não levou em conta desabilidades físicas, motoras ou visuais, mas apenas a não disponibilidade do ambiente ou objeto para o uso por uma pessoa, sem a necessidade de um terceiro para auxiliar. A desabilidade de uma pessoa pode ser vista não como uma característica inerente do indivíduo em sim, mas como produto da interação deste com o ambiente que o cerca (RIMMER; ROWLAND, 2008).

A falta de acessibilidade em ambientes domésticos tem sérios impactos nas pessoas que dele dependem. Problemas de saúde relacionados a ausência de prática regular de exercícios, e maus hábitos alimentares são muito mais frequentes em pessoas portadoras de desabilidades (RIMMER; ROWLAND, 2008). O aumento da complexidade de execução de tarefas simples como estocagem de alimentos e preparação de refeições, e o cansaço gerado por este acréscimo de complexidade são fatores para para que as atividades físicas sejam menos frequentes.

Além do impacto na saúde física, ambientes que tornam pessoas dependentes de outras para tarefas corriqueiras, ou que apenas dificultam estas tarefas, tem impacto na saúde psicológica das pessoas que neles vivem. Ter de viver em um ambiente não adaptado deteriora o senso de segurança e identidade, principalmente em idosos. Além de que ao desempenhar suas tarefas diárias em ambientes sem acessibilidade, portadores de necessidades especiais convivem com um risco muito maior de quedas, entre outros acidentes domésticos (SIXSMITH; SIXSMITH, 2008).

Entre a população idosa, é muito maior o tempo passado em casa, em relação às populações mais jovens. A maioria dos idosos com mais de 75 anos preferem viver em suas próprias casas, a viver com parentes, ou casas de repouso. Além disso, o número de idosos vivendo sozinhos em suas próprias casa é crescente. Desta maneira, o lar da pessoa idosa passa a ter grande importância para que esta tenha um envelhecimento saudável (PORTET et al., 2013).

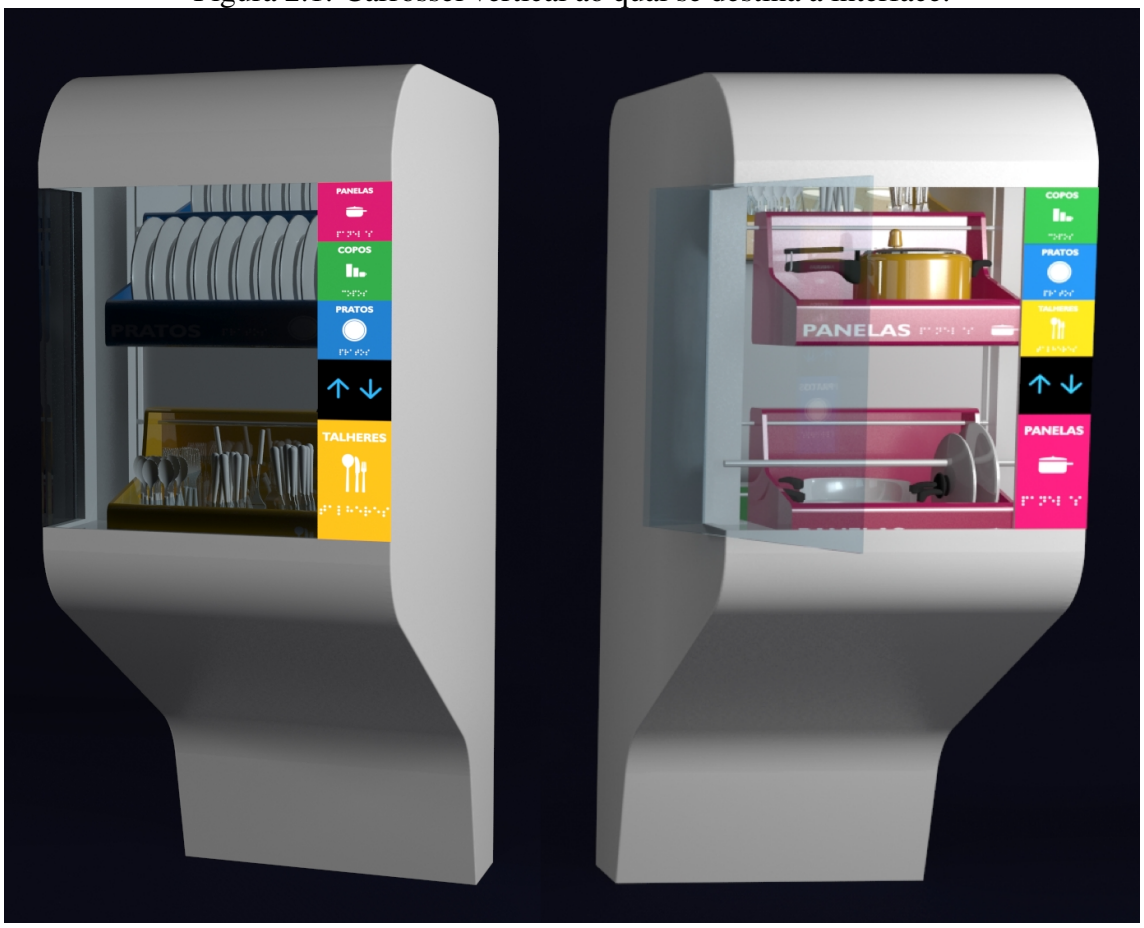
Com o envelhecimento da população, a preocupação para que as pessoas tenham um envelhecimento saudável em suas próprias casa passou a ser uma preocupação real. Tendo a Organização Mundial da Saúde definido este termo como o nível de bem estar, autonomia e participação social de que a pessoa dispõe nesta fase da vida (BICKENBACH, 2011).

2.2 Prateleira automatizada

Em (ARNOLD; KLENNER, 2015) é apresentado o projeto de uma estante para o auxílio de pessoas idosas. Neste trabalho os autores buscaram elaborar um móvel doméstico para a cozinha, que fosse totalmente adaptado para o uso por idosos. O móvel, representado na figura 2.1, consiste de um carrossel vertical de prateleiras, acionado por motores. As prateleiras são classificadas de acordo com sua finalidade, entre pratos, talheres, e copos. Através de um conjunto de botões posicionado à direita do móvel, o usuário pode escolher a prateleira que será disposta ao alcance deste. A prateleira atual, a qual será referência de posicionamento durante o trabalho, é a que fica disposta mais abaixo, ao alcance das mão do usuário.

Um carrossel vertical serve para armazenar itens diversos num conjunto de prateleiras móveis, sendo utilizado principalmente em ambientes industriais. Seu funcionamento, como mostrado na figura 2.2, se dá pela movimentação das prateleiras verticalmente. Isso permite que o usuário alcance o objeto de sua escolha sem ter que subir até a

Figura 2.1: Carrossel vertical ao qual se destina a interface.



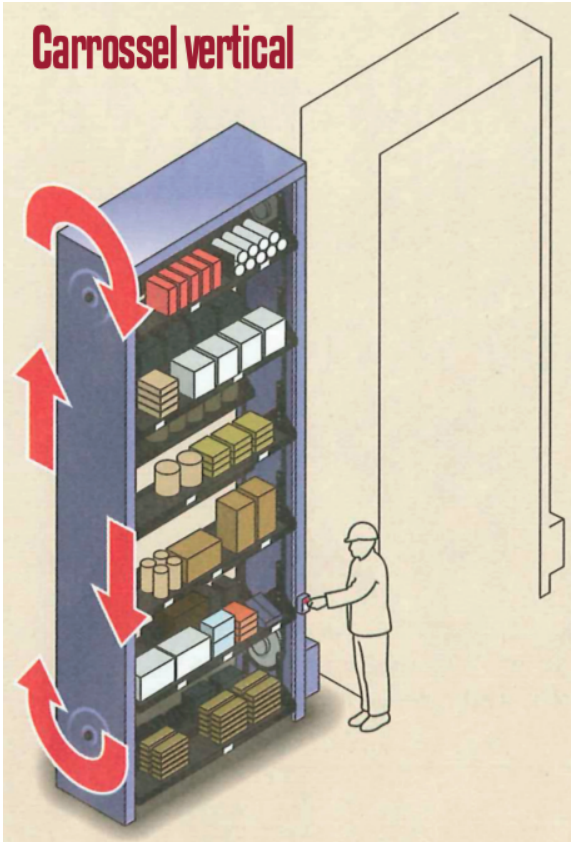
Fonte: (ARNOLD; KLENNER, 2015)

prateleira que o contém, trazendo-a até si.

As decisões de projeto feitas pelos autores seguiram o objetivo de dar acessibilidade a idosos na cozinha, diminuindo os riscos de acidentes domésticos, e de se colocar em posições não ergonômicas, ao buscar itens em armários, estantes e afins. Para isto os autores tiveram cuidado de não empregar materiais cortantes no projeto, e calcular dimensões que tornem o acesso a estante confortável para idosos cadeirantes, ou com outras restrições de mobilidade.

Observando a funcionalidade e objetivo do trabalho dos autores, percebeu-se uma oportunidade de estender o uso do dispositivo a outros ambientes doméstico, além da cozinha, para onde foi projetado. Para isto algum tipo de interface acessível poderia ser concebida, o que levou à pesquisa em direção às interfaces de voz. Após leituras sobre o assunto, observou-se a importância que interfaces de voz bem definidas podem ter para dar acessibilidade a usuários de tecnologia de *Smart Homes*. E mais que isso, observou-se uma oportunidade de agregar em uma única interface características de interfaces de voz observadas como positivas em trabalhos esparços, e validar, através de testes com

Figura 2.2: Funcionamento de um carrossel vertical.



usuários a eficácia destas características quando empregadas juntas.

3 OBJETIVOS

Nesta seção são apresentados os objetivos gerais do trabalho no âmbito de validar a aceitação de aspectos de interfaces de voz voltada ao controle de um móvel doméstico adaptado à acessibilidade. E objetivos específicos, de implementar uma arquitetura para o desenvolvimento e manutenção de uma interface de voz em um dispositivo móvel. E de definir um conjunto de comandos de voz fácil de utilizar, com base na preferência dos usuários, e obtidos através de testes prévios com os mesmos.

3.1 Objetivos finais

Após pesquisa sobre interfaces de voz, e suas aplicações em dispositivos, observou-se que diferentes interfaces implementam diferentes maneiras de expor ao usuário as possibilidades de interação com esta, para dar comandos ao dispositivo que se quer utilizar. Como descrito em (KAMM, 1995), os usuários trazem expectativas quando na comunicação com interfaces de voz, baseadas na sua própria expertise na comunicação falada. Assim usuários diferentes terão expectativas diferentes a serem atingidas.

Este trabalho foi desenvolvido com o objetivo final de avaliar a aceitação de um conjunto de características observadas como positivas por usuários de diferentes interfaces de voz testadas, quando reunidas numa mesma interface. Assim como avaliar se a presença concomitante destas características é de maneira geral vantajosa, ou se isto pode degradar a experiência do usuário.

3.2 Objetivos específicos

Como observado por (COLEMAN et al., 1994), é uma preocupação constante na indústria manter alta a manutenibilidade do código das aplicações. Isto pois estima-se que o custo de manutenção de um software durante seu ciclo de vida pode chegar a mais da metade do seu custo de desenvolvimento.

Baseando-se nisso, os objetivos específicos deste trabalho são adaptar uma arquitetura de código já existente para implementar a interface de voz a que o trabalho refere-se. Arquitetura cujos requisitos são a manutenção simplificada, que permita a inclusão, remoção e alteração de contextos da interface. E que mantenha o esforço de implementação

constante ao longo de seu desenvolvimento.

4 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo trataremos sobre a pesquisa que embasou o andamento do trabalho. O trabalho se iniciou com estudo sobre acessibilidade, e o papel da área de *Smart Homes* na resolução de problemas observados. Em seguida o estudo se direcionou a interfaces de voz, seu papel como ferramenta de inclusão de pessoas com desabilidades. Chegando então a trabalhos elaborados na área, onde interfaces de voz foram utilizadas em aplicações reais.

4.1 *Smart homes* e acessibilidade

A busca por tornar ambientes mais acessíveis, ou disponíveis, aos seus usuários, impulsionou a pesquisa na área de *Smart Homes* (PORTET et al., 2013). Entre outras motivações, uma delas foi a busca de adaptações que facilitassem a interação de portadores de desabilidades com seus lares e objetos do dia-a-dia. Ao estudo de tecnologias voltadas a este fim, deu-se o nome de *Assistive Homes* (NUSSBAUM, 2006).

Tecnologia assistiva é o nome dado a criação de dispositivos que melhoram as capacidades físicas e sensoriais de pessoas com deficiência, com o intuito de ajudá-las a viver de maneira mais independente (DEWSBURY; EDGE, 2001). O avanço da tecnologia no sentido de gerar autonomia em pessoas tem impacto na qualidade de vida destas. Desenvolver dispositivos de acessibilidade práticos e eficazes não promovem só a integração da pessoa com o ambiente, mas a aceitação de novas tecnologias auxiliares no seu dia-a-dia (PORTET et al., 2013).

As facilidades trazidas por este tipo de tecnologia direcionada aos portadores de necessidades especiais é útil também ao público geral. Assim, as soluções de acessibilidade para ambientes domésticos devem ser pensadas para o uso por pessoas, independente de suas habilidades ou limitações (DEWSBURY; EDGE, 2001). E ainda, sendo tecnologia de auxílio, os dispositivos desenvolvidos não devem substituir completamente a atuação da pessoa no ambiente, pois isto pode contribuir para um estilo de vida não saudável (PORTET et al., 2013).

4.2 Interfaces de voz

Uma interface de voz é aquilo com o que uma pessoa interage ao se comunicar com um sistema de linguagem falada. São partes da interface (COHEN et al., 2004):

- As mensagens de sistema, sendo estas pré-gravadas, ou geradas com voz sintética em tempo real;
- A gramática, ou o conjunto de sentenças aceito na interface a cada momento;
- A lógica de diálogo, que é o conjunto de ações da interface em resposta aos comandos recebidos do usuário. Estas ações incluem tanto a atuação do sistema em seus periféricos, quanto na mudança de estado e contexto da própria interface.

4.2.1 Vantagens da automação por voz

Do ponto de vista do usuário convencional, a tecnologia de controle por voz trás vantagens em relação às interfaces gráficas habituais por diversas questões (ROSENFELD; OLSEN; RUDNICKY, 2001). A possibilidade do usuário interagir com uma interface, sem que esta ocupe suas mãos, ou exija que este despenda seu foco visual totalmente para ela, permite a execução de tarefas simultâneas. Pela natureza omnidirecional do som, alertas sonoros podem ser identificados, mesmo quando o usuário não está atento ao sistema (LINES; HONE, 2006).

Já para usuários com necessidades especiais, interfaces de voz trazem um ganho considerável em acessibilidade (LINES; HONE, 2006). Por não requerer capacidades manuais, pessoas com limitações físicas ou motoras podem interagir com este meio sem prejuízos. Na situação de pessoas acamadas, a não direcionalidade da comunicação permite operar a interface sem a necessidade de trocar de posição (RASHID et al., 2017). Como a incidência de problemas de visão é alta entre a população idosa, a possibilidade de operar sistemas sem necessitar atenção visual trás inclusão a estas pessoas, assim como a deficientes visuais em geral.

4.2.2 Importância de interfaces bem definidas

Uma interação entre humano e máquina é dita bem sucedida quando a comunicação entre ambos é completada de maneira simples e eficiente no consumo de tempo, do

ponto de vista do humano envolvido (KAMM, 1995). Interfaces de voz são dependentes temporais, assim, diferentemente de interfaces gráficas, para se apresentar uma maior quantidade de informações ao usuário, consome-se tempo deste, e não espaço em um monitor de vídeo. Somando-se a isso a ausência de referências espaciais, é possível observar a importância de se definir bem uma interface de voz, para que a interação com o usuário não seja desgastante, ou temporalmente ineficiente.

Como a fala é um meio de comunicação natural, podemos aproveitar a expertise do usuário nesta área para tornar mais eficiente as interações deste com a interface. Mas ao mesmo tempo que estas habilidades natas são úteis, a expertise trás expectativas ao usuário. Desta maneira o sucesso na interação com interface vai depender da satisfação destas expectativas inerentes à comunicação falada.

Trabalhos como (YANKELOVICH; LEVOW; MARX, 1995a) e (KAMM, 1995) descrevem boas práticas, apontadas como vantajosas em testes com usuários, a serem levadas em conta na definição de interfaces de voz, .

4.2.2.1 Adaptabilidade à expertise dos usuários

O protocolo da comunicação falada torna o usuário incapaz de desacoplar 3 comportamentos de suas reações durante a interação com a máquina: a tendência a antecipar as frases da máquina, e a tendência em falar junto com a máquina, mesmo sabendo que estas ações possam dificultar a comunicação com o sistema.

Estes comportamentos ocorrem principalmente pelo impulso de atalhar os caminhos conhecidos. Numa conversa normal entre duas pessoas, isto demonstra ao interlocutor que determinada parte do assunto é conhecida, e ele pode avançar mais rapidamente na comunicação. Para evitar que o usuário sinta-se preso à cadência de fala da máquina, ou aos fluxos de comunicação quando este já compreende bem seus funcionamentos, deve-se permitir o ajuste da velocidade das falas, assim como a omissão da informação não mais necessária.

4.2.2.2 Voz e sons

De forma geral a voz natural é preferível à voz sintética, gerada por computador. Mas observou-se que sons icônicos, e notas sonoras são muito eficazes para auxiliar na navegação pelos fluxos da interface. Sons icônicos permitem o usuário inferir um contexto complexo da aplicação, sem que mais informação seja passada a este. Já notas sonoras

podem organizar a comunicação, indicando momentos de fala do usuário e da máquina, ou ainda, deixar claro ao usuário de que o sistema está executando alguma ação, e logo retornará, evitando silêncios desconfortáveis.

4.2.2.3 Diálogos

Um cuidado especial é necessário quando ocorrem diálogos, com perguntas e respostas, entre interface e usuário. Nestes momentos a falta de prosódia da voz sintética pode interferir no entendimento de questões pelo usuário. Frases claras e diretas são importantes nestes contextos. O uso e entendimento de nomes próprios também pode gerar problemas de comunicação por má interpretação ou má pronúncia dos mesmos. E em relação ao vocabulário e linguagem utilizados, deve-se levar em conta que os usuários têm dificuldade em manter a comunicação falada utilizando linguagem mais formal.

4.2.2.4 Navegação

Para obter as intenções do usuário em contato com a interface, é comum interfaces de voz questionarem constantemente o usuário sobre sobre cada dado e cada aspecto a todo o momento. Isto torna artificial a interação com a interface. Por isso deve-se buscar deixar o usuário tomar a iniciativa sobre os aspectos que lhe são mais importantes a cada momento. O uso de contextos de utilização na aplicação permite que o usuário saiba a cada momento com quais aspectos deve interagir para manter o controle sobre o sistema. Além disso, é sempre importante direcionar o tipo de resposta esperada de um usuário ao se fazer uma pergunta, para que este formule suas frases com segurança, e diminuam-se falhas de comunicação.

4.2.2.5 Comandos

Por não ser dependente espacial, as interfaces de voz podem disponibilizar um número virtualmente ilimitado de comandos a cada momento, diferentemente de interfaces gráficas, que são limitadas pelo espaço dos monitores empregados. Por outro lado, os comandos expostos pela interface são armazenados na memória de curta duração do usuário, sendo então limitados pela quantidade de comandos que o usuário consegue armazenar em cada momento. Então o número e a complexidade dos comandos dispostos ao usuário devem ser levados e conta.

Sem a referência visual, notou-se que naturalmente o usuário passa a usar refe-

rências temporais, para se referir a itens, índices e listas. Comandos utilizando palavras como "anterior/próximo", "antes/depois", "seguinte"são automaticamente buscados pelos usuários.

4.2.2.6 Feedbacks, avisos e lembretes

Por não ter todos os aspectos da aplicação visualmente monitorados a todo o momento, o usuário de interface de voz tende a sentir menos controle sobre a aplicação. Esta falta de segurança pode ser diminuída com o uso de avisos, lembretes e *feedbacks*.

Avisos podem servir para informar automaticamente o usuário de que algum aspecto da aplicação se encontra em estado que exige atenção. Apenas o fato de saber que esta informação será passada sem que o usuário tenha que questionar sobre ela, é suficiente para diminuir as preocupações durante o uso.

Os lembretes têm o papel de mostrar ao usuário quais aspectos da aplicação merecem sua atenção a cada momento, e tiram a necessidade do usuário decorar todas as informações de um mesmo contexto, para poder questionar sobre este. *Feedbacks* são importantes, pois mantém o usuário informado do entendimento da máquina sobre seus comandos. Retornar sons e falas que identificam como os comandos foram interpretados permitem que o usuário identifique falhas de comunicação, e as corrija antes que acarretem erros.

4.2.2.7 Menus globais e de ajuda

Disponibilizar ao usuário um menu com comandos globais, que possam ser acessados em qualquer contexto da aplicação, permite ao usuário encurtar caminhos de navegação, além de dar a sensação de estar em um ambiente multitarefa. Pois estar em um contexto de aplicação, e poder atalhar para interferir em aspectos de outro contexto cria a ideia de que diferentes contextos estão ativos simultaneamente no sistema, e disponíveis ao usuário.

Menus de ajuda são menus globais especializados em dar informações gerais sobre a aplicação, e específicas sobre o contexto atual, como uma lista dos comandos aceitos. Verificou-se que as pessoas se apoiam neste tipo de ajuda constantemente, até atingirem determinado nível de expertise na interface que estão utilizando.

4.3 Arquitetura

Arquitetura em uma aplicação é a divisão em alto nível das partes que compõe esta aplicação, de acordo com suas funcionalidades e interações mútuas. Tratar sobre arquitetura de software é decidir sobre questões base de organização, que no caso de necessidade futura, uma mudança qualquer poder ser bastante custosa (FOWLER, 2002).

4.3.1 Padrões de design

O processo de definir a arquitetura da aplicação passa pela etapa de definir e delimitar seus componentes, pensando no objetivo de cada um. Por ser um processo exaustivamente reproduzido na indústria, algumas formas de organização acabam sendo repetidamente utilizadas, sendo por sua eficiência ao delimitar a tarefa a ser cumprida pelos componentes, ou por simplificar o entendimento do componente desenhado. Estas formas mais utilizadas acabam se tornando padrões de design, os quais acabam sendo difundidos e melhorados por serem repetidamente reproduzidos e avaliados pelos usuários (GAMMA et al., 1993).

O uso de padrões de design trás duas vantagens principais ao processo de design da aplicação (GAMMA et al., 1993).

4.3.1.1 Reusabilidade

Uma característica de padrões de design é ter sua finalidade bem definida, o que permite aplicar uma mesma solução em diferentes aplicações que necessitem de funcionalidades afins. Ao desenvolver um componente seguindo um padrão, o código escrito poderá ser reutilizado indefinidamente. E a cada manutenção recebida tende, este a melhorar sua qualidade e confiabilidade.

4.3.1.2 Cobertura de casos

Ao desenhar um componente, o designer de software deve pensar nos diversos casos de uso para este, e nas situações adversas às quais ele pode ser submetido. Ao utilizar um padrão de design na arquitetura, além da funcionalidade que se espera do componente, o designer tem ainda a vantagem de estar cobrindo casos adversos observados e contornados por quem desenvolveu o padrão, sem precisar sequer conhecer estes casos.

Optou-se pelo uso de padrões de design na arquitetura da interface de voz, que é objeto deste trabalho, pelas vantagens e facilidades que estes trazem ao desenvolvimento de software. Observando que a plataforma *iOS* de dispositivos móveis foi escolhida para o desenvolvimento, foram pesquisados padrões de design comuns no desenvolvimento de aplicativos de celular. As subseções a seguir versarão sobre estes padrões de design, que possuem em comum o objetivo de separar o comportamento de interfaces dos dados que estas buscam representar.

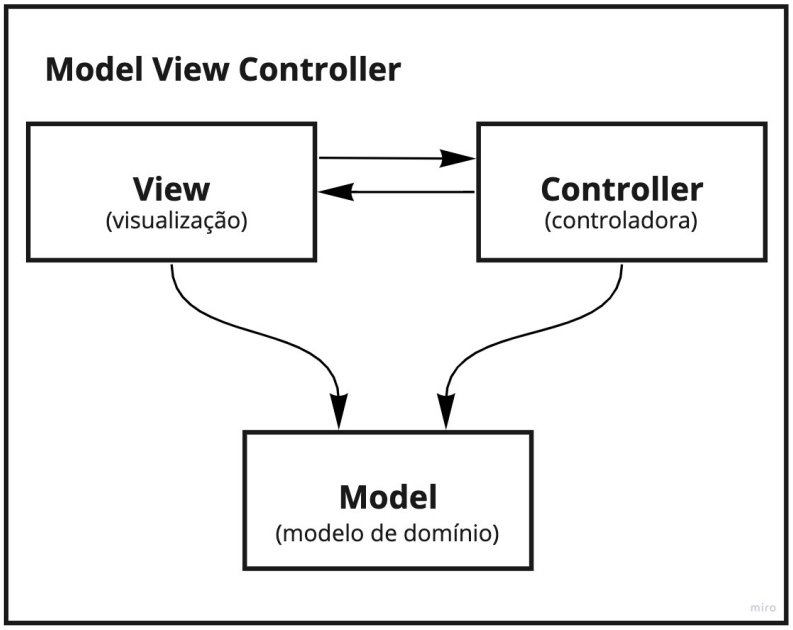
4.3.2 *Model View Controller*

Um dos padrões de design que se tornou referência no desenvolvimento de interfaces de usuários foi o *Model View Controller* (MVC). Uma arquitetura que implemente o padrão *Model View Controller* estará definindo uma separação clara entre três papéis (QURESHI; SABIR, 2014), como observados na figura 4.1, que em ambientes onde se pratica a orientação a objetos podem ser descritos por:

- ***Model***, que é o **modelo**, representado por um objeto do domínio, contendo os dados que a aplicação pretende trabalhar, sendo ao apresentar ao usuário, ou modificá-los de acordo com a regra de negócio (FOWLER, 2002). O modelo define o estado da aplicação, e é responsável por atender as requisições da visualização, entregando a esta dados atualizados.
- ***View***, que é a **visualização**, servindo à disposição da informação do modelo ao usuário. Seu papel é apenas representar os dados, sem modificá-los (FOWLER, 2002). A visualização é responsável por renderizar os dados, e manter-se atualizada sempre que notificada sobre atualizações do modelo.
- ***Controller***, que é a **controladora**, tendo três tarefas principais. A interpretação dos comandos dados pelo usuário. A atualização dos dados do modelo, de acordo com a regra de negócio. E o disparo das atualizações da visualização, para que esta represente a cada momento a versão mais recente dos dados do modelo (FOWLER, 2002). (QURESHI; SABIR, 2014).

A separação entre modelo e visualização é útil para separar duas preocupações importantes: a de desenhar a melhor visualização e representação de dados para usuário em cada contexto, e a preocupação de manter os dados da aplicação consistentes com as regras de negócio. Esta separação permite que diversas visualizações representem

Figura 4.1: Estrutura do padrão MVC.



Fonte: O autor, 2019

o mesmo conjunto de dados, cada uma à sua maneira, com a vantagem de que basta atualizar um modelo para que todas estas visualizações atualizem suas representações. E mais importante, ao separar modelo e visualização garante-se que mudanças em qualquer visualização de determinado modelo não implica em mudanças no modelo em si, ou em suas regras de negócio (FOWLER, 2002).

Já a separação entre visualização e comportamento é interessante para os casos em que os métodos de interação com o usuário podem mudar em uma mesma visualização. Diferentes métodos de introdução de comandos, ou diferentes permissões de interação podem ser aplicados à mesma visualização, sem a necessidade de atualizar a visualização em questão.

O diagrama representado na figura 4.1, mostra que há um isolamento do modelo, o qual está imune tanto a modificações da visualização, quanto da controladora. Em compensação, tanto visualização quanto controladora dependem diretamente do modelo: a controladora por observar modificações no estado representado pelo modelo para disparar modificações na visualização, e a visualização por observar e apresentar diretamente os dados do modelo. Isso culmina na situação em que qualquer modificação mais profunda da estrutura do modelo, acarreta em modificações tanto da controladora, quanto da visualização. Além de a controladora e a visualização dependerem diretamente do modelo, há uma dependência mútua entre controladora e visualização, o que mantém os módulos fortemente acoplados. Este acoplamento não chega a ser um grande problema em aplicações *desktop* com fluxos simples, onde o número de canais de interação do usuário com a

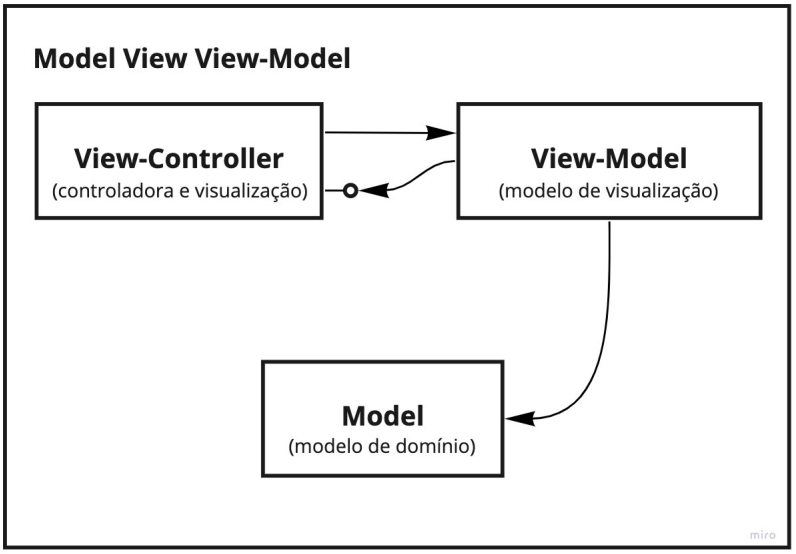
aplicação é baixo. Mas em ambientes web, onde são comum os fluxos mais complexos, e onde tanto as entradas de comandos, quanto as variações de representação dos dados são maiores, o padrão pode se tornar um problema.

4.3.3 *Model View View-Model*

Com a popularização do padrão de design MVC, observou-se que em situações onde apenas uma controladora era designada a uma visualização, a divisão clara entre visualização e controladora não era tão vantajosa em comparação à separação entre visualização e modelo do domínio. Nestes casos os desenvolvedores passaram a juntar à visualização a lógica da controladora. Em paralelo a isso passou a ficar evidente a dificuldade da controladora gerenciar o estado imediato da visualização, nos casos em que este é independente do domínio em si. Estados temporários resultantes de interações do usuário com a visualização, que não acarretavam modificações do domínio, não tinham um local para serem corretamente definidos (SYROMIATNIKOV; WEYNS, 2014).

Neste ambiente idealizou-se o padrão de design *Model View View-Model* (MVVM). Este padrão incluiu um quarto componente ao padrão originário MVC: o *View-Model*, ou modelo da visualização. Em compensação reconheceu-se a vantagem de unir as lógicas da visualização e da controladora em um único componente: a *View-Controlle*r. Na composição de três componentes, as atribuições podem ser vistas como:

Figura 4.2: Estrutura do padrão MVVM.



Fonte: O autor, 2019

- ***View-Controller*** é a união entre as lógicas da **visualização** e da **controladora**. Dentro da *View-Controller*, a visualização manteve suas atribuições de represen-

tar os dados do modelo, e ser atualizada ao comando da controladora, que se encontra junta no mesmo módulo. Do mesmo modo a controladora, dentro da *View-Controller*, manteve suas atribuições de interpretar os comandos do usuário, e consultar e atualizar os dados do modelo. A diferença entre *View-Controle*r, e os componentes de visualização e controladora do padrão MVC está no modelo consumido e atualizado, que deixou de ser o modelo do domínio diretamente, passando a ser o modelo de apresentação, o *View-Model*.

- ***View-Model*** é o modelo de apresentação. O modelo de apresentação é uma camada entre o modelo do domínio e a *View-Controller*. Este é responsável principalmente por gerar um modelo de dados pronto para ser representado pela *View-Controller*, sem que a última precise adaptá-lo de qualquer forma. Este modelo próprio de dados agrega tanto os dados do domínio necessário à visualização, quanto os dados do estado local da visualização que não teriam relação nenhuma com o domínio. E por último passou a ser papel do modelo de apresentação a atualização dos dados do modelo de domínio. Deste modo, mesmo ao reunir visualização e controladora num mesmo elemento, o modelo de apresentação impede que a lógica do domínio tenha contato com a lógica de visualização.
- ***Model*** permanece sendo o modelo de domínio. O modelo de domínio manteve seu papel de apenas representar o domínio, ignorando a existência das lógicas de visualização e controladora, e agora, a lógica do modelo de apresentação.

O padrão MVVM manteve o objetivo principal de separação entre lógica de domínio e lógica de visualização, idealizados no padrão MVC que o originou. Em relação ao padrão MVC incluiu-se o modelo de apresentação para gerenciar o estado da visualização. Isto permitiu juntar lógicas de visualização e controladora mantendo desacoplados visualização e modelo de domínio. Esta última união de lógicas é possível no padrão de design, mas não essencial, podendo visualização e controladora manterem seus papéis separados.

Como podemos observar na figura 4.2, a *View-Controller* depende apenas do modelo de apresentação, mas não existe a dependência inversa. A *View-Controller* avisa diretamente o modelo de apresentação sobre as interações do usuário. Ao perceber que o estado da visualização precisa ser alterado, o modelo de apresentação notifica indiretamente a *View-Controller*, para que esta observe o estado atual, e atualize a visualização contida em si. E quando as interações requerem atualização dos dados do domínio, o modelo de apresentação dá a este o comando de atualizar, e se preciso, os dados necessários

para tal.

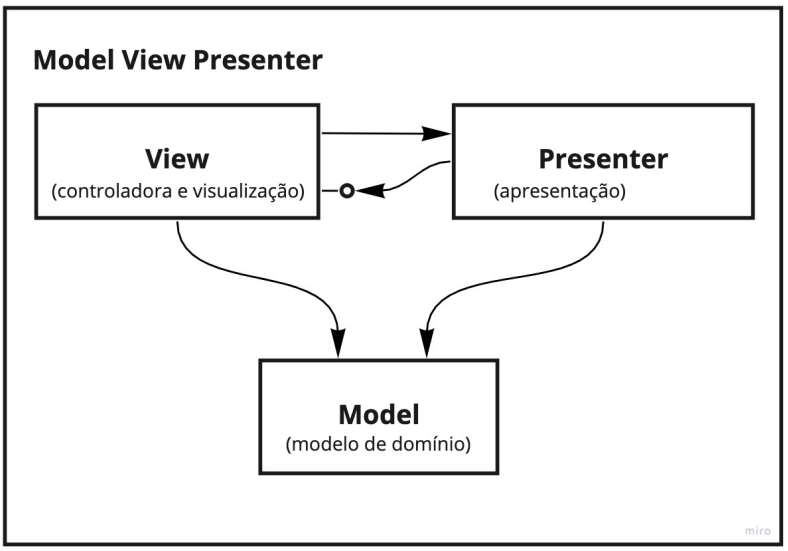
4.3.4 *Model View Presenter*

A busca por separar a lógica da visualização da lógica do modelo foi o incentivo para a idealização do padrão de design MVC. O interesse em manter esta separação, unido à busca por um desacoplamento entre controladora e modelo de domínio levou ao desenvolvimento de um padrão mais flexível, o MVVM. Este padrão trouxe o desacoplamento desejado, mas reduziu a flexibilidade do acesso direto da visualização à leitura do modelo de domínio. Avaliando este cenário, desenvolveu-se o padrão *Model View Presenter* (MVP). Nesta estrutura foi devolvida à visualização o acesso aos dados do domínio, ao mesmo tempo que se manteve a capacidade de renderizar dados não presentes no domínio, uma vez que estes compõem o estado da visualização, que fica armazenado neste novo componente, o *Presenter*. A divisão de responsabilidades entre os módulos podem ser descritas como segue:

- **Model**, que segue representando o modelo de domínio da aplicação.
- ***View***, que se aproxima da *View-Controller* do padrão MVVM, ao unir as lógicas da visualização e da controladora. Mas no padrão MVP este componente retomou seu acesso direto aos dados do modelo de domínio, que no MVVM era restrito aos dados contidos no modelo de apresentação. Esse acesso direto ao domínio deu mais flexibilidade ao padrão de design. Com isto a visualização recuperou sua autonomia de buscar diretamente no domínio a informação a ser representada, dependendo de um intermediário apenas para verificações de regras de negócio, as quais esta segue desconhecendo.
- ***Presenter***, ou apresentação, é uma versão mais enxuta do modelo de apresentação observado no padrão MVVM. A apresentação tem o papel de observar diretamente o domínio, e avisar a visuaização sobre atualizações nos dados deste. Mas ao contrário do modelo de visualização, a apresentação não lê dados do domínio para entregá-los à visualização, ela apenas notifica a visualização, e esta atualiza-se. A apresentação é responsável por interceptar e gerenciar os comandos do usuário, atualizando tanto o modelo de domínio, quanto o estado da visualização quando necessário. O estado da visualização, que não tem relação com o estado do modelo de domínio, fica armazenado na apresentação. E uma diferença entre o modelo de

apresentação do MVVM e a apresentação do MVP é a capacidade da apresentação modificar componentes da visualização diretamente. Isto pois a apresentação possui referências a estes componentes, podendo diretamente atualizar parte da visualização ao perceber alguma modificação no modelo do domínio.

Figura 4.3: Estrutura do padrão MVP.



Fonte: O autor, 2019

Ao mesmo tempo que a *View-Controller* do MVVM é acrescida de lógica para lidar diretamente com os dados do modelo de domínio, para se tornar a *View* do padrão de design MVP, esta perde a lógica de atualização dos campos que dependem apenas do seu estado local, que independe do modelo de domínio. Assim, como pode ser observado na figura 4.3, retoma-se a dependência entre apresentação e visualização. Esta dependência permite uma visualização mais passiva, que expõe seus componentes para atualização por parte da apresentação.

4.3.5 Considerações sobre a escolha do padrão de design

Após os estudos sobre padrões de design para arquiteturas de aplicações com interfaces para usuários, optou-se pelo uso do padrão *Model View View-Model*. A decisão foi baseada na plataforma escolhida para a implementação: *iOS*. Por ser direcionada a dispositivos móveis, a interação entre visualização e controladora é bem definida, e pré estabelecida nas ferramentas de desenvolvimento da plataforma. Com isto o uso de uma *View-Controller*, que una visualização e controladora, torna-se simples, e é um facilitador para o desenvolvimento. Outro ponto são os recursos de de sintetização e reconhecimento de voz, e reprodução de áudio, que serão consumidos pela controladora e visualização da

aplicação. Como estes recursos serão consumidos constantemente em contextos diferentes, ao utilizarmos o modelo de apresentação do MVVM, ao invés da apresentação do MVP, podemos tirar a complexidade de fazer o modelo do domínio se conformar a visualização da aplicação.

4.4 Trabalhos relacionados

Em (IPIÑA; LORIDO; LÓPEZ, 2011), o autor propõe um sistema de navegação e reconhecimento de produtos para ser aplicado em supermercados. O sistema é desenvolvido em uma aplicação para *smartphone* e faz uso de identificação por radiofrequência para localizar corredores e prateleiras e guiar o usuário na sua locomoção. Já para identificar determinado produto, faz-se o uso de um código de barras bidimensional, o qual deve ser reconhecido pela câmera do aparelho celular em uso. Uma interface de voz é empregada para fazer buscas por produtos da loja, utilizando nomes de itens cadastrados. O usuário pode escolher entre o modo de identificação de produto e o modo de navegação, utilizando comando de voz, ou através de gestos na tela do *smartphone*.

(STIFELMAN et al., 1993) apresenta um sistema para organização de notas mentais, direcionado ao uso através de interface voz. O sistema é implementado em dispositivo móvel, que apresenta uma serie de comandos de fala para classificar e buscar notas mentais, as quais também são armazenadas por comandos de fala. Não fazendo nenhum uso de interface gráfica, não é disponibilizado *feeback* visual de texto reconhecido. Para usuários mais experientes, é possível acelerar a taxa de fala da interface, tornando a interação mais rápida. O sistema é preemptível no momento de fala da maquina, permitindo que o usuário interrompa a fala da interface de voz para fazer sua requisição, sem que isto acarrete em prejuízo ao reconhecimento dos comandos dados.

Em (YANKELOVICH; LEVOW; MARX, 1995a) o autor descreve o *SpeechSystem*, que é uma ferramenta para a criação de interfaces de voz. O sistema foi desenvolvido para testar abordagens diversas e fazer verificações junto a usuários de como diferentes aspectos de uma interface de voz interferirem positivamente ou negativamente na experiência do usuário ao interagir com esta. No trabalho publicado, o sistema é testado com um conjunto de tarefas rotineiras, como responder *e-mails* e observar compromissos em uma agenda. O sistema usa comandos pré-estabelecidos, que podem ser chamados pelo usuário utilizando sentenças diferentes. A obtenção destas sentenças foi feita através de pesquisa prévia com possíveis usuários, questionando que frases estes elaborariam para

dar comandos a tal sistema. Há a implementação de *feedbacks* sonoros e visuais do estado do sistema e da execução de tarefas. Assim como o texto reconhecido é apresentado visualmente ao usuário. O sistema também aborda recuperação de falhas, auxiliando o usuário a falar seus comando utilizando as sentenças corretas, e de maneira clara.

(PORTET et al., 2013) trata sobre o design e validação de uma interface de voz voltada para o uso em *Smart Home*, pensando em pessoas idosas. O sistema captura constantemente o som do interior do ambiente, buscando por comandos feitos para controlar aspectos deste ambiente, como acendimento de luzes e ajuste da temperatura. A interface utiliza processamento de linguagem natural, diferentemente da maioria das interfaces, que empregam comandos pré-definidos. O uso de linguagem natural torna o reconhecimento de comandos muito mais custoso para o sistema, mas ao mesmo tempo permite uma comunicação mais fluida e natural com o usuário. O sistema dispõe de botões para auxiliar os comandos, e um monitor para *feedback* visual do estado do sistema, e dos comandos de voz reconhecidos.

Em (O'NEILL et al., 1999) o autor apresenta um robô comandado por um interface de voz, que auxilia pessoas com desabilidades visuais a se locomover no ambiente. O usuário segura os manipuladores do robô, enquanto dá comandos de deslocamento, e este move-se pelo ambiente contornando obstáculos, e informando ao usuário através de fala sobre tais obstáculos.

Em (YASUMURA; YOSHIDA; YOSHIDA, 2006) apresenta-se um controle remoto para ar-condicionado desenvolvido para cegos. O controle usa uma interface de voz para permitir que o usuário ajuste os parâmetros de um ar-condicionado através de comandos de voz. Isto tira a necessidade dos deficientes visuais de decorar a posição de cada botão do controle remoto. O controle recebe os comandos do usuário, e quando perguntando, informa o estado atual da configuração do ar-condicionado.

O autor, em (MOIR et al., 2008), apresenta uma interface de voz para controlar uma *Smart Home*. Para definir os comandos desta interface, propositalmente utilizou-se um vocabulário simples e limitado, no intuito de tornar o conjunto de comandos mais facilmente memorizável. O sistema utiliza um monitor de vídeo para apresentar um avatar na forma de um rosto humano, para dar *feedbacks* visuais sobre a fala reconhecida, e sobre a prontidão a receber comandos do usuário. Implementou-se também um menu de ajuda, acessível em qualquer contexto da aplicação, o qual utiliza uma voz diferente da voz principal, deixando claro para o usuário que este se encontra no contexto do menu de ajuda.

Em (HAMILL et al., 2009) apresenta um sistema que faz uso de uma interface de voz para pessoas enfermas, ou em situação vulnerável que passam períodos sozinhas em casa. O objetivo é permitir que, no caso de uma emergência, a pessoa possa acionar uma central de socorro utilizando apenas comandos de voz. O sistema monitora a pessoa em casa utilizando microfones, buscando sons incomuns, que possam indicar uma queda, ou outro tipo de acidente doméstico. Ao perceber uma variação em relação aos padrões do ambiente, o sistema dispara a interface de voz, a qual tem o papel de questionar o usuário e identificar a necessidade de se chamar uma equipe de socorro automaticamente. O sistema utiliza frases gravadas, baseadas nos protocolos de atendimentos das centrais de emergência canadenses, para dar sensação de familiaridade e segurança ao usuário. Além disso, as frases são personalizadas, sendo gravadas utilizando o nome do usuário em questão para chamá-lo, e a ele se referir. Para facilitar a comunicação, e tornar o processo tão rápido quanto o possível, o sistema utiliza a interface de voz para fazer perguntas cujas as respostas são direcionadas a sim ou não. Com isso evitam-se falhas de interpretação de voz, e possíveis erros que estas podem acarretar.

Em (BILMES et al., 2005), o autor apresenta uma interface de voz voltada para sistemas que precisam de comandos contínuos. Os comandos não se baseiam em linguagem falada, mas em aspectos e características do som. A interação do usuário se dá através de sons contínuos, e as variações de altura, frequência, intermitência e volume definem continuamente variações nas entradas do sistema controlado. Além de sons contínuos, uma serie de sons discretos também são mapeados para estímulos e comandos imediatos.

O presente trabalho apresenta o processo de pesquisa, caracterização e desenvolvimento de uma interface de voz para controlar um carrossel vertical de uso doméstico. Da mesma forma que (STIFELMAN et al., 1993) e (IPIÑA; LORIDO; LÓPEZ, 2011), a implementação foi realizada sobre plataforma de dispositivos móveis, mas diferente destes e das proposta apresentadas por (HAMILL et al., 2009) e (BILMES et al., 2005), a interface desenvolvida apresenta *feedbacks* sonoros sobre a aceitação e rejeição de comandos, bem como o estado e contexto em que o usuário se encontra em cada momento.

Diferente das propostas descritas em (BILMES et al., 2005), (HAMILL et al., 2009), (MOIR et al., 2008) e (STIFELMAN et al., 1993), a interface desenvolvida implementa uma interface gráfica auxiliar, que mostra o contexto em que o usuário se encontra, cada palavra reconhecida durante a fala do usuário, e um botão para que o usuário ativamente defina o momento em que irá emitir comandos à maquina. Da mesma forma que (IPIÑA; LORIDO; LÓPEZ, 2011), o sistema como um todo tem o intuito de catalogar

objetos para depois poder buscá-los. Mas com a diferença de utilizar apenas comandos de voz para tal, necessitando o uso de códigos de barras bidimensionais atrelados aos objetos. Na usabilidade, a tarefa de catalogar e buscar itens se aproxima da implementada em (STIFELMAN et al., 1993).

5 PROPOSTA

Neste capítulo apresentaremos a proposta de implementação da interface de voz que é objeto deste trabalho. Aqui serão descritos os contextos de navegação disponíveis ao usuário e sua organização hierárquica, as características observadas como vantajosas na seção 4.2.2 e que foram elencadas para implementação na interface, e a arquitetura proposta para organizar o desenvolvimento do software.

5.1 Escolha da plataforma da aplicação

Foi escolhida a plataforma iPhone, sobre o sistema operacional *iOS* de dispositivos móveis da *Apple*, para dar suporte à interface de voz pelos seguintes motivos:

- Dispositivos móveis são presentes no dia-a-dia das pessoas. Isto trás familiaridade no uso de aplicações quando desenvolvidas para estes dispositivos.
- O sistema operacional *iOS* dá acesso às ferramentas de interpretação de texto em voz sintética, presentas na biblioteca *AVFoundation*. Assim como à ferramenta de transcrição de frases faladas em texto, presente na biblioteca *Speech*. Ambas de amplo emprego na implementação da interface de voz objeto deste trabalho.
- O desenvolvimento de aplicações para esta plataforma utiliza a linguagem *Swift*, a qual já é dominada pelo autor, facilitando a implementação.
- A utilização da interface via dispositivo móvel tira a necessidade de adicionar dispositivos de processamento de voz ao projeto da prateleira proposto em (ARNOLD; KLENNER, 2015).

5.2 Implementando boas práticas

Como visto em (YANKELOVICH; LEVOW; MARX, 1995a) e (KAMM, 1995), podemos elencar boas práticas de desenvolvimento de interfaces de voz já apontadas como positivas em trabalhos similares. Do conjunto de boas práticas estudado, selecionamos algumas para a implementação da interface objeto deste trabalho. Aqui listaremos seus empregos:

- **Clareza dos diálogos:** Evitou-se o uso de perguntas por parte da interface para

o usuário, isto para contornar a falta de prosódia da voz sintética disponibilizada pelo sistema operacional *iOS*. Ao passo que as informações passadas ao usuário são específicas, não dando espaço para interpretações.

- **Número de comandos reduzidos:** O fluxo de organização foi organizado em contextos. Cada um destes contextos abrigam um número reduzido de comandos, o que permite que o usuário decore um pequeno conjunto de comandos a cada momento durante a navegação.
- **Voz:** Foi feito o uso de voz sintética para replicar as falas da interface. Isto permite que novos comandos e mensagens sejam gerados automaticamente facilitando manutenções futuras e o próprio desenvolvimento da interface.
- **Sons:** Foram implementados ícones de áudio para indicar as mudanças de contexto durante a navegação, assim como para indicar sucesso ou falha de comandos dados pelo usuário, ou ações tomadas pela prateleira. Para toda ação, ou navegação que possui uma fala descritiva atrelada, um ícone de áudio específico daquela ação ou navegação é reproduzido, logo antes da fala.
- **Adaptação à expertise do usuário:** Para os usuários iniciantes, a interface toca todas as frases informativas sobre o estado da interface, e sucesso ou falha de comandos e ações. Para usuários acostumados com a interface, é possível suprimir as frases que anunciam a transição entre contextos da aplicação, o que torna um pouco mais fluida a navegação. Já para usuários experientes é possível remover todas as falas da interface, com exceção das contidas no contexto de ajuda. Deste modo o usuário será guiado unicamente pelos ícones de áudio, que não sofrerão alterações.
- **Áudio reconhecido:** Uma interface gráfica simples foi incluída na implementação para informar textualmente toda a fala reconhecida pela interface de voz, permitindo ao usuário detectar possíveis erros de interpretação dos comandos falados.
- **Avisos:** A interface avisa por linguagem falada se cada comando do usuário foi aceito ou rejeitado, bem como o sucesso ou falha de cada ação requisitada pelo usuário.
- **Menus Global e de Ajuda:** Foram incluídos na implementação um menu global de comandos, e um menu de ajuda. O menu global é acessível de qualquer contexto de navegação, e contém os comandos de cancelar a ação atual, o que faz o usuário retornar ao contexto imediatamente anterior ao atual. O comando de cancelar e voltar ao início, o qual além de cancelar a ação atual, encaminha o usuário ao contexto

inicial da aplicação. E contém também o comando de ajuda, o qual leva o usuário diretamente ao contexto de ajuda. O contexto de ajuda informa ao usuário o que o contexto atual tem a oferecer ao usuário, ou o que a interface espera de entrada por parte dele. Os textos sintetizados no contexto de ajuda são falados mais pausadamente, e utilizando uma voz levemente mais grave, para que o usuário tenha pleno sentido de que se encontra num contexto específico de ajuda.

5.3 Comandos

Para o controle da estante de prateleiras automatizada, descrita em (ARNOLD; KLENNER, 2015), foi concebido um conjunto de comandos com o intuito de estender o uso para além do ambiente da cozinha, para o qual foi projetada. Os comandos visam gerenciar os itens nas prateleiras, assim como as próprias prateleiras e o espaço para inserir novos objetos, que estas apresentam. Há ainda os comandos de busca, que tem por efeito mover as prateleiras da estante, e levar até o usuário o item buscado.

Como a interface não possui processamento de linguagem natural, os comandos são frases pré estabelecidas, que devem ser ditas pelo usuário. Como o reconhecimento é o puro casamento de texto falado, com os textos pré-estabelecidos, optou-se por definir diversas variações de frases para cada comando, para que o usuário possa usar a variação mais confortável à sua fala. Há também frases com variações pequenas, como a presença ou não de artigos, preposições, etc... Deste modo um mesmo comando tem mais chances de ser reconhecido no caso de o usuário fazer modificações na maneira de formulá-lo verbalmente.

O conjunto completo de comandos é descrito abaixo, dividido nos contextos de navegação, os quais serão elucidados na seção seguinte:

- **Cancelar:** Cancela a ação atual, retornando ao contexto imediatamente anterior ao do início da ação. Não havendo ação em andamento, apenas retorna ao contexto anterior. Este comando faz parte do menu global, e pode ser chamado de qualquer ponto do fluxo de navegação.
- **Cancelar e voltar ao início:** Cancela a ação em andamento, e retorna ao contexto inicial da aplicação. Não havendo ação em andamento, apenas retorna ao contexto inicial. Este comando faz parte do menu global, e pode ser chamado de qualquer ponto do fluxo de navegação.

- **Ajuda:** Encaminha o usuário ao contexto de ajuda. Este comando faz parte do menu global, e pode ser chamado de qualquer ponto do fluxo de navegação.
- **Listar comandos:** Somente acessível no contexto de ajuda. Lista os comandos do contexto imediatamente anterior ao do pedido de ajuda, e imediatamente leva o usuário a este contexto anterior. Caso a interface não aguarde comandos, informa qual ação é esperada do usuário no momento.
- **Alterar experiência do usuário:** Aguarda o usuário informar um valor de expertise entre iniciante, intermediário e experiente. A interface ajustará o número de frases faladas por esta ao nível de experiência indicado, e encaminhará o usuário para o contexto anterior. Caso o usuário informe um valor de experiência não listado, ele é avisado do erro, e um novo valor é solicitado.
- **Verificar presença de item:** Aguarda o usuário informar o nome do item a verificar, e informa se o item se encontra presente em alguma prateleira da estante ou não. Caso se encontre, informa também a sua quantidade. Em seguida retorna ao contexto anterior.
- **Adicionar item à prateleira atual:** Aguarda o usuário dar um nome ao item a ser guardado. Este deve dizer o nome do item, e inseri-lo na prateleira atualmente exposta. A aplicação guarda uma lista de palavras protegidas, as quais não podem ser usadas como nomes para objetos ou prateleiras. Se o nome dado não estiver contido nessa lista, o objeto é inserido na lista de objetos da prateleira atual, e o usuário é encaminhado ao contexto anterior. Caso o nome seja rejeitado, o usuário é informado, e um novo nome é solicitado.
- **Adicionar lista de itens à prateleira atual:** Aguarda o usuário dar um nome ao primeiro objeto da lista a ser guardado. Se o nome não estiver entre as palavras protegidas da aplicação, o objeto é inserido na lista de itens da prateleira atual, o usuário é informado do sucesso, e a interface aguarda pelo próximo item da lista. Caso o nome seja rejeitado, o usuário é informado da rejeição, e um novo nome é solicitado.
- **Fim da lista de inserção:** Comando dado para finalizar a inserção de uma lista de itens. Com este comando a inserção da lista se encerra, e o usuário é encaminhado ao contexto imediatamente anterior.
- **Remover item da prateleira atual:** Aguarda o usuário informar o nome do item a remover. Se o item estiver presente na prateleira atual, este é removido da lista de

itens da prateleira. Caso o item não se encontre presente na prateleira atual, o usuário é informado da ausência, e encaminhado ao contexto imediatamente anterior.

- **Remover lista de itens da prateleira atual:** Aguarda o usuário informar o nome do primeiro item da lista de remoção. Se o item estiver presente na prateleira atual, este é removido dela, o usuário é informado, e a interface passa a aguardar o nome do próximo item da lista de remoção. Caso o item não se faça presente na prateleira atual, o usuário é informado, e a interface passa a aguardar o nome do próximo item a ser removido.
- **Fim da lista de remoção:** Comando dado para finalizar a remoção de uma lista de itens. Após este comando a inserção é encerrada, e o usuário é encaminhados para o contexto imediatamente anterior.
- **Atualizar status do espaço da prateleira atual:** A interface passa a esperar pela informação de espaço a ser passada pelo usuário, que deve ser um valor entre prateleira cheia ou prateleira com espaço. Não é permitido ao usuário informar prateleira vazia, pois este estado é atingido apenas com a remoção de todos os itens constantes da prateleira. Ao informar um status de espaço válido, o usuário é encaminhado ao contexto imediatamente anterior. No caso de um status inválido, o usuário é informado da rejeição, e a interface aguarda a informação de um status válido.
- **Status do espaço da prateleira atual:** A interface informa o status de espaço da prateleira atual. Após, retorna ao contexto anterior.
- **Renomear prateleira atual:** A interface passa a aguardar pelo novo nome para a prateleira atual. Caso o usuário informe um nome que não tenha sido usado em outra prateleira, e não figure entre as palavras protegidas da aplicação, a prateleira atual é renomeada, e o usuário é encaminhado ao contexto anterior.
- **Próxima prateleira:** Comando de efeito imediato. Move a prateleira atual para baixo, fazendo com que a prateleira imediatamente acima ocupe seu lugar, e figure como prateleira atual.
- **Prateleira anterior:** Comando de efeito imediato. Move a prateleira atual para cima, fazendo com que a prateleira imediatamente abaixo ocupe seu lugar, e figure como prateleira atual.
- **Buscas:** Encaminha o usuário ao contexto de buscas.
- **Listagens:** Encaminha o usuário ao contexto de listagens.
- **Buscar prateleira pelo nome:** Aguarda o nome da prateleira a buscar. Caso o

usuário informe o nome ou o número de uma prateleira existente, a prateleira é trazida até ele, ocupando o local da prateleira atual. Caso contrário, o usuário é informado da ausência, e é encaminhado ao contexto anterior.

- **Buscar objeto pelo nome:** A interface aguarda o nome do item a buscar. Caso o usuário informe o nome ou o número de um objeto presente em qualquer prateleira, a prateleira que contém o objeto é trazida até ele, ocupando o local da prateleira atual. Caso contrário, o usuário é informado da ausência, e é encaminhado ao contexto anterior.
- **Buscar prateleira cheia:** Havendo prateleira cheia, esta é trazida ao usuário, ocupando o local da prateleira atual. Caso contrário, o usuário é avisado da ausência e encaminhado ao contexto anterior.
- **Buscar prateleira vazia:** Havendo prateleira vazia, esta é trazida ao usuário, ocupando o local da prateleira atual. Caso contrário, o usuário é avisado da ausência e encaminhado ao contexto anterior.
- **Buscar prateleira com espaço:** Havendo prateleira com espaço, esta é trazida ao usuário, ocupando o local da prateleira atual. Caso contrário, o usuário é avisado da ausência e encaminhado ao contexto anterior.
- **Listar itens na prateleira atual:** Lista os itens presentes na prateleira atual, e suas quantidades. Ao fim o usuário é retornado ao contexto imediatamente anterior.
- **Listar itens de uma prateleira:** Aguarda o nome da prateleira a listar. Caso o usuário informe o nome, ou o número de uma prateleira existente, o conteúdo desta é listado, e ao fim o usuário é encaminhado ao contexto imediatamente anterior. Caso contrário, a ausência da prateleira é informada, e a interface aguarda por um nome de prateleira válido.
- **Listar itens totais da estante:** Lista todos os itens presentes na estante, e suas quantidades. Ao fim o usuário é encaminhado ao contexto imediatamente anterior.
- **Listar prateleira presentes:** Lista o nome das prateleiras presentes, utilizando os nomes dados pelo usuário, ou, na ausência destes, os número originais destas.

A interface sempre informa sobre a aceitação ou rejeição de todo o comando e entrada de informação pelo usuário. Tanto uma mensagem de aceitação ou rejeição é falada, quanto um ícone de áudio é tocado.

5.4 Contextos: Agrupamento de comandos

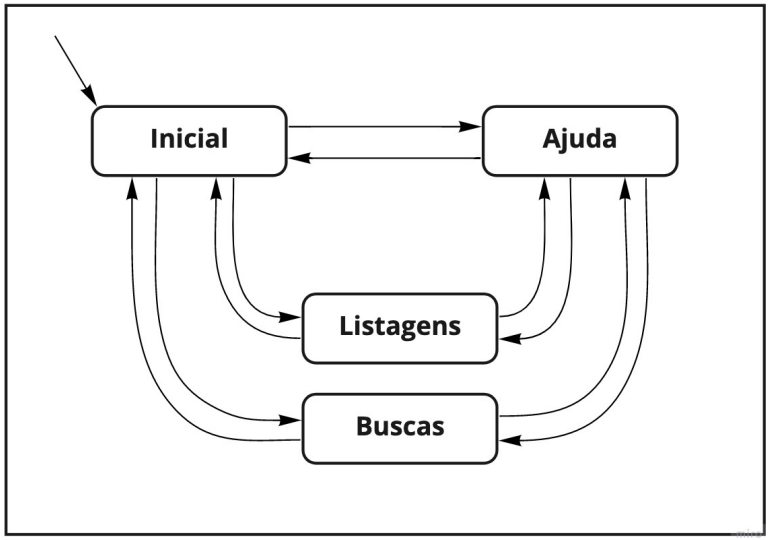
Ao passo que interfaces gráficas levam em conta restrições espaciais para a exibição de conteúdo ao usuário, interfaces de voz trabalham com restrições temporais. Uma aplicação gráfica simples pode ter um grande número de comandos e informações representadas numa mesma tela simultaneamente, aumentando a sensação e controle do usuário durante sua operação.

Já em interfaces de voz o número de comandos e informações passíveis de serem apresentados ao usuário a cada momento é reduzido drasticamente, devido à necessidade do usuário memorizar as possibilidades de interação a todo momento. Se mesmo em interfaces gráficas há a necessidade de se organizar um fluxo de interação em diferentes telas com diferentes contextos, por não ser viável expor o usuário a todas as informações a todo o momento, em interfaces de voz esta necessidade é ainda mais urgente.

Para o usuário melhor entender as interações às quais é exposto num determinado momento, é importante que estas tenham alguma relação semântica. Em termos práticos, se um usuário sabe o assunto objeto de determinada interface, em determinado momento, este estará mais apto a buscar comandos e informações que façam sentido no contexto da aplicação em que este se encontra. Além disso, o próprio agrupamento de informações semanticamente relacionadas, durante o desenho de interfaces, torna mais fácil o entendimento das regras de negócio da aplicação por parte do usuário (PARK; LEE, 2011).

Em uma interface de voz, como a tratada neste trabalho, é importante observar que o usuário não tem acesso constante e visual aos gatilhos de comandos e informações que lhes podem ser úteis na tarefa que esse está desempenhando. Isso pois tudo é transmitido ao usuário oralmente, e não é viável a utilização de uma interface que informe constantemente as opções que estão disponíveis ao usuário a cada momento. Dessa forma, o agrupamento de comandos relacionados se torna essencial para definir contextos de interação semanticamente coesos. Pois uma vez que para descobrir a lista de comandos que se pode dar à interface, o usuário terá que ouvir da interface a descrição de cada comando. É essencial que o número de comandos seja reduzido, tanto para economizar tempo do usuário, quanto para a memorização eficaz das opções. Além disso, a proximidade semântica entre os comandos dá um significado ao contexto, e facilita que o usuário memorize um conjunto maior de comandos, e seja capaz de supor com mais precisão a existência de outros comandos quando não memorizados (KAMM, 1995) (YANKELOVICH; LEVOW; MARX, 1995b).

Figura 5.1: Hierarquia de contextos e navegação.



Fonte: O autor, 2019

Com base no que foi exposto, os comandos definidos em 5.3 serão agrupados por proximidade semântica em quatro grandes contextos.

5.4.1 Contexto Inicial

O contexto inicial é o principal do fluxo de navegação, e por esse motivo, uma de suas funcionalidades é direcionar o usuário aos outros contextos da aplicação quando solicitado, como observado na figura 5.2. Este contexto é direcionado principalmente às interações com a prateleira atualmente exposta ao usuário. Os comandos nele abrigados, relacionados ao seu valor semântico, são:

- Adicionar um item à prateleira atual;
- Adicionar uma sequência de itens à prateleira atual;
- Remover um item da prateleira atual;
- Remover uma sequência de itens da prateleira atual;
- Verificar o espaço da prateleira atual;
- Renomear a prateleira atual;
- Verificar a presença de um item nas prateleiras;
- Avançar para a próxima prateleira;
- Retornar para a prateleira anterior;
- Definir se prateleira cheia ou com espaço para mais itens;

- Ir para o contexto de busca;
- Ir para o contexto de listagens;

O contexto inicial apresenta a seguinte descrição ao usuário quando solicitada ajuda no contexto:

"Você está no contexto inicial. Aqui você pode interagir com a prateleira atualmente exposta, ou ir para os contextos de buscas, ou listagens."

5.4.2 Contexto de Buscas

No contexto de buscas o usuário tem contato com os comandos de busca da aplicação. Busca por objetos, por prateleiras específicas, ou por espaço para guardar mais objetos na estante. Os comandos de busca acessíveis através deste contexto são:

- Buscar prateleira pelo nome;
- Buscar prateleira com espaço para mais objetos;
- Buscar prateleira vazia;
- Buscar prateleira cheia;
- Buscar item por;

O contexto de buscas apresenta a seguinte descrição, quando solicitada ajuda:

"Você está no contexto de buscas. Aqui você pode buscar por objetos, ou prateleiras. Caso o item em questão esteja presente, ele será trazido até você."

5.4.3 Contextos de Listagens

No contexto de listagem o usuário tem acesso ao inventário da prateleira, podendo listar os itens presentes em cada prateleira, todos itens presentes na estante, ou os nomes atuais das prateleiras da estante. O acesso a estas informações se dá através do seguintes comandos:

- Listar todos os itens da estante;
- Listar itens de uma prateleira;
- Listar prateleiras;
- Listar itens da prateleira atual;

O contexto de listagens apresenta a seguinte descrição, quando solicitada ajuda:

"Você está no contexto de listagem. Aqui você pode listar itens e prateleiras presentes."

5.4.4 Contexto de Ajuda

O contexto de ajuda é acessível a partir de qualquer contexto em que o usuário se encontre. Nele o usuário pode pode obter informações sobre o objetivo do contexto em que se encontra atualmente, e principalmente, quais os comandos aceitos no contexto e que ações a interface espera do usuário. Ao entrar neste contexto, o usuário recebe uma breve descrição de onde se encontra no fluxo de navegação. O contexto possui dois comandos:

- Informar comandos;
- Mais informações;

Se o usuário solicitar ajuda, estando no contexto de ajuda, este receberá a seguinte mensagem:

"Você está no contexto de ajuda. Você pode pedir mais informações sobre os contextos da aplicação, ou pedir pela listagem de possíveis comandos dizendo apenas: comandos"

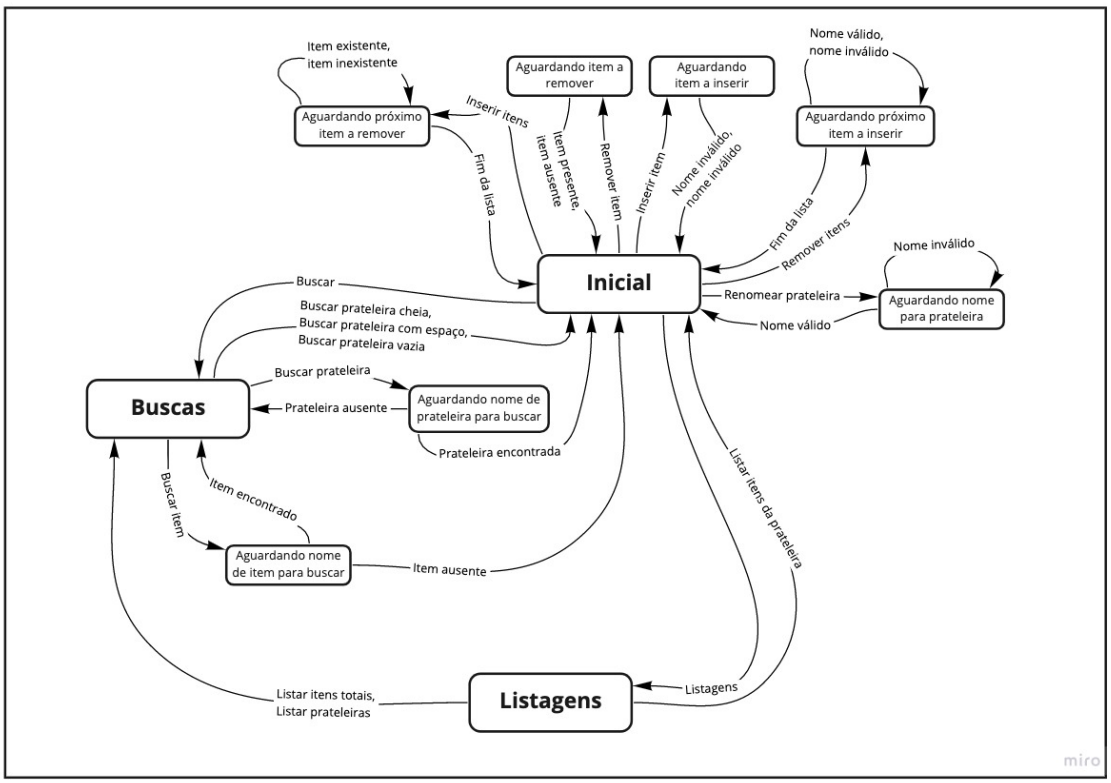
5.5 Arquitetura proposta

Nesta seção trataremos da arquitetura da aplicação, expondo o padrão de design utilizado, e adaptado às necessidades da aplicação. Expondo também as classes especializadas para a implementação do padrão. Também trataremos de maneira geral sobre os módulos que compões a aplicação, e como se organizam.

5.5.1 Adaptando o *Model View View-Model*

Como especificado e justificado na seção 4.3.5, o padrão de design escolhido para guiar a implementação da interface será o *Model View View-Model*. A biblioteca de desenvolvimento de interfaces gráficas *UIKit* disponibilizada pela *Apple* para desenvolvimentos

Figura 5.2: Diagrama de comandos que geram trocas de contexto.
os comandos.jpg



Fonte: O autor, 2019

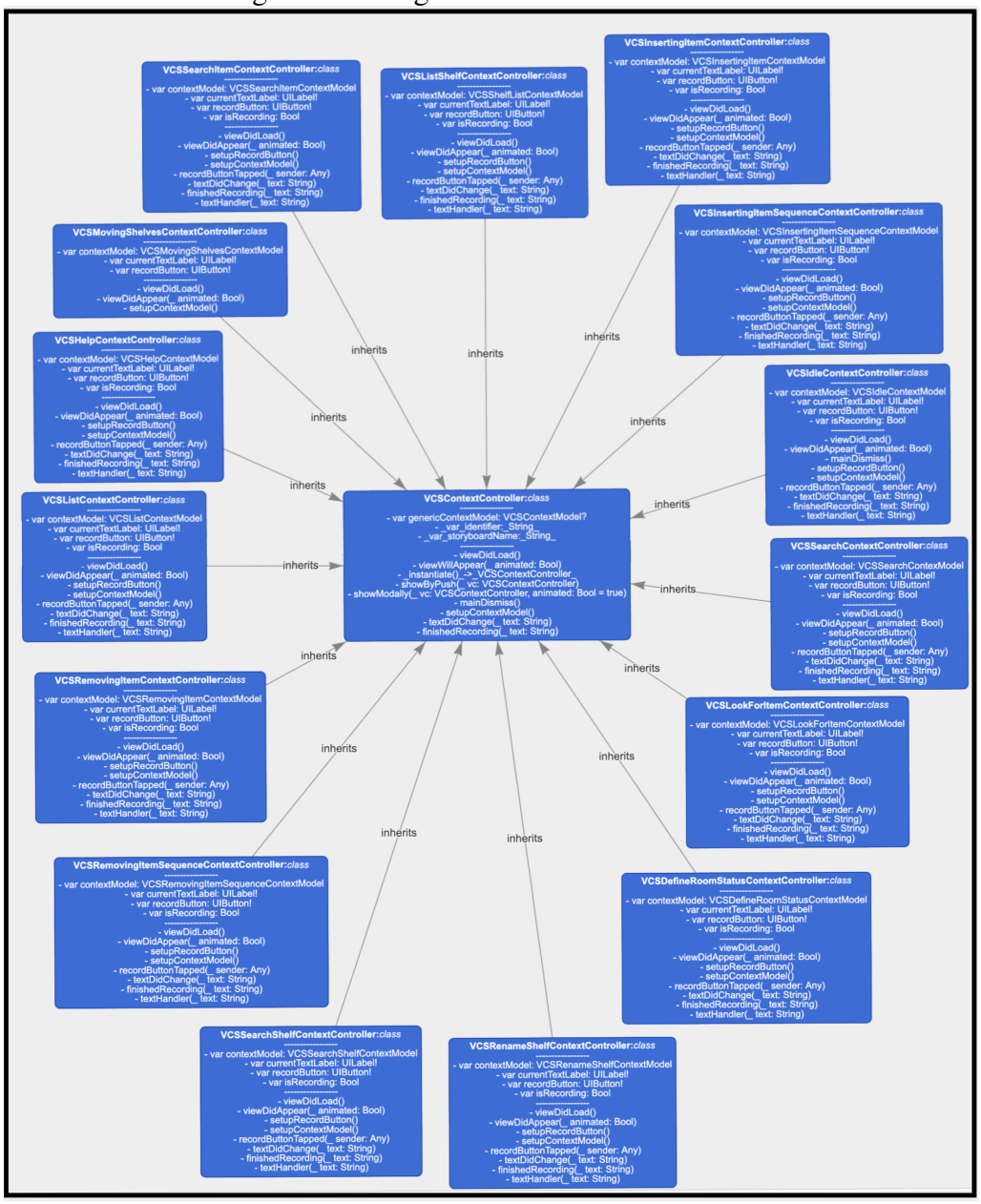
de aplicativos para o sistema operacional *iOS* oferece a classe *UIViewController*, abriga tanto a visualização, quando a controladora, sendo responsável por interceptar os comandos do usuário, e passá-los ao modelo de visualização. Bem como sendo responsável por atualizar a informação apresentada ao usuário, de acordo com os conteúdos do modelo de visualização.

5.5.1.1 VCSContextController

Para suprir as necessidade do projeto, a classe *UIViewController* foi especializada, criando-se a classe *VCSContextController*. A sigla VCS está para *Voice Controlled Shelf*. A modificação da palavra *View* para *Context* deu-se pelo novo intuito da classe, de servir como controladora não apenas da interface gráfica, mas também da interface por comando de voz, onde cada instância representará um contexto do fluxo de navegação. Como controladora de contexto, esta classe tem o papel de utilizar o serviço de reconhecimento de voz para receber comandos de voz do usuário. Da mesma maneira que a controladora no padrão de design MVVM, a *VCSContextController* será avisada pelo seu modelo de visualização, ou modelo de contexto, como chamaremos para esta adaptação de arquitetura,

de que os dados do contexto foram atualizados. Mas como os dados a serem apresentados ao usuário são também na forma de texto falado, e ícones de áudio, esta controladora de contextos será responsável por direcionar os dados de saída ao usuário, através do serviço de sintetização de voz, e do serviço de execução de ícones de áudio, que serão tratados mais a frente. Outra responsabilidade da controladora de contexto é efetuar a navegação do usuário entre os contexto, carregando na memória o próximo contexto quando necessário, e descartando a si mesmo quando comandado a retornar ao contexto imediatamente anterior.

Figura 5.3: Diagrama das classes de controladoras



Fonte: O autor, 2019

5.5.1.2 VCSContextModel

A classe *VCSContextModel* foi criada como modelo de visualização do padrão MVVM que desejou-se implementar. Por servir como modelo de dados para a controladora de contextos, recebeu este nome para sua nova semântica de modelo de contexto. Este modelo de contexto é responsável por avisar sua controladora quando deve apresentar informações ao usuário. No caso de textos a serem falados, estes ficam guardados de forma textual para serem sintetizados em voz, a cargo da controladora de contextos. Já os ícones ficam guardados pelo seus nomes. Além disso, o modelo de contexto também é responsável por ler os dados do domínio, e repassá-los à controladora de contexto para sua exibição. Assim como também é responsável por atualizar os dados de domínio quando solicitado pela controladora de contexto. A comunicação do modelo de contexto à controladora de contexto sobre informações a apresentar é feita por atributos do tipo função, chamados reatores, que ficam armazenados no modelo de contexto. Há um reator designado para cada situação a ser comunicada à controladora de contexto.

5.5.1.3 Reatores

Reatores são atributos que podem guardar trechos de código, que guardam consigo o contexto do ambiente de execução em que foram instanciados (PROKOPEC, 2018). Na aplicação utilizada neste trabalho, o modelo de contexto possui estes atributos, os quais são inicializados pela controladora de contexto, através de atribuição direta de um bloco de comando. No momento oportuno, o modelo de contexto irá executar este bloco de comandos, na forma de uma reação a determinado estímulo recebido.

Após sua inicialização, a controladora do contexto de listagem inicializa em seu modelo de contexto, e atribui a cada atributo reator deste um trecho de código específico ao que este reator se destina. O código carregado no reator correspondente ao comando de listagem de prateleiras corresponde a uma função com um parâmetro do tipo texto. Esta função contém uma chamada ao método de sintetização de texto em voz, do serviço de sintetização de voz. E o parâmetro de texto do reator é passado como parâmetro de texto a ser sintetizado. Como é possível observar no diagrama de sequência da figura 5.5, ao receber um comando do usuário, a controladora do contexto de listagem comunica-se com seu modelo de contexto, através de uma chamada ao método deste que corresponde ao comando de listagem reconhecido. Neste método o modelo de contexto busca no modelo de domínio a informação da lista das prateleiras presentes, a qual é retornada de forma

textual. Ao receber a lista em forma de texto, o modelo de contexto executa o código de seu reator correspondente ao comando de listagem de prateleiras. Como o código ali armazenado pela controladora de contexto é uma chamada ao método de sintetização de voz a partir de texto, do serviço de sintetização de voz, ao ser executado o reator gera uma chamada de método serviço de sintetização de voz por parte da controladora de contexto. O método de sintetização de texto em voz recebe a lista de prateleira de forma textual, a transforma em áudio falado, que é executado para o usuário ouvir.

5.5.1.4 Controladoras e modelos de contexto da aplicação

A classe *VCSContextController* e todas suas subclasses são preparadas para tocar um ícone de áudio e falar uma breve descrição do contexto que cada subclasse representa. Tanto ícone de áudio, quanto frases de descrição ficam armazenados nos seus modelos de visualização, herdados de *VCSContextModel*, e possuem estas propriedades herdadas da subclasse. O total dos modelos de visualização, que herdam de *VCSContextModel* pode ser visto no diagrama da figura 5.4. Já o total das controladoras implementadas, que herdam de *VCSContextController* pode ser observado no diagrama da figura 5.3. Abaixo segue a lista com as descrições de cada par controladora/modelo de visualização especializados apartir das classes *VCSContextController* e *VCSContextViewModel* respectivamente:

- ***VCSIdelContextController* e *VCSIdleContextViewModel*:** Responsáveis por interpretar os comandos do contexto inicial, executar suas ações e encaminharem os usuários para os contextos seguintes.
- ***VCSInsertingItemContextController* e *VCSInsertingItemContextViewModel*:** Responsáveis por gerenciar o processo de inserção de item, recebendo e avaliando o nome do item a ser inserido, e reagindo de acordo com a aceitação ou rejeição do nome.
- ***VCSContextRemovingItemController* e *VCSContextRemovingItemViewModel*:** Responsáveis por gerenciar o processo de remoção de item, recebendo e avaliando o nome do item a ser removido, e reagindo de acordo com a presença ou ausência do item na estante de prateleiras.
- ***VCSInsertItemSequenceContextController* e *VCSInsertItemSequenceContextViewModel*:** Responsáveis por gerenciar o processo de inserção de uma sequência de itens, recebendo e avaliando o nome de cada item a ser inserido, e reagindo de

acordo com a aceitação ou rejeição de cada nome, bem como ao comando de fim de lista.

- ***VCSRemoveItemSequenceContextController* e *VCSRemoveItemSequenceContextViewModel*:** Responsáveis por gerenciar o processo de remoção de uma sequência de itens, recebendo e avaliando o nome do item a ser removido, e reagindo de acordo com a presença ou ausência de cada item, bem como ao comando de fim de lista.
- ***VCSRenameShelfContextController* e *VCSRenameShelfContextViewModel*:** Responsáveis por gerenciar a renomeação da prateleira atual. Avaliando a validade do nome dado, e reagindo de acordo com sua aceitação ou rejeição.
- ***VCSLookForItemContextController* e *VCSLookForItemContextViewModel*:** Responsáveis por gerenciar o processo de busca por um item, recebendo um nome do usuário e procurando por um item com este nome em todas as prateleiras. Após a busca, informando sobre a presença ou ausência do item nomeado.
- ***VCSDefineRoomStatusContextController* e *VCSDefineRoomStatusContextViewModel*:** Responsáveis por gerenciar o processo de definição do status de espaço de uma prateleira, recebendo a informação de espaço, e avaliando sua validade. Alterando o status do espaço se necessário. E, se necessário, informando a invalidade do status recebido, pedindo para informar novamente na sequência.
- ***VCSSearchContextController* e *VCSSearchContextViewModel*:** Responsáveis por gerenciar os comandos do contexto de buscas interpretando e executando suas ações de busca.
- ***VCSSearchShelfContextController* e *VCSSearchShelfContextViewModel*:** Responsáveis por gerenciar o processo de busca por prateleiras, avaliando a presença do nome buscado, e trazendo a prateleira, se presente, até o usuário.
- ***VCSSearchItemContextController* e *VCSSearchItemContextViewModel*:** Responsáveis por gerenciar o processo de busca por um objeto, avaliando a presença do nome em uma prateleira, e trazendo a prateleira que contém o objeto até o usuário, se o objeto estiver presente na estante.
- ***VCSListContextController* e *VCSListContextViewModel*:** Responsáveis por gerenciar os comandos do contexto de listagens, e listando os objetos conforme solicitado.
- ***VCSListShelfContextController* e *VCSListShelfContextViewModel*:** Responsáveis

por gerenciar o processo de listagem dos itens de uma prateleira, avaliando a validade do nome de prateleira passado pelo usuário, informando sobre sua inexistência, ou informando seu conteúdo no caso da prateleira estar presente.

- ***VCSMovingShelvesContextController* e *VCSMovingShelvesContextViewModel*:** Responsáveis por se comunicar com os controles da estante física, se presente, e por informar o usuário sobre o início, andamento e sobre o fim do processo de movimentação das prateleiras, bem como atualizar o estado do posicionamento das prateleiras, representado pelo nome da prateleira atualmente exposta ao usuário, no gerenciador de inventário.
- ***VCSHelpContextController* e *VCSHelpContextViewModel*:** Responsáveis por gerenciar os comandos do contexto de ajuda, listando os comandos utilizando a voz sintetizada com parâmetros de voz e entonação específicos do contexto de ajuda quando requisitado.

5.5.2 Serviços de entrada e saída

Além de uma interface gráfica simplificada, como saída, e um botão nesta interface que opera como entrada, a aplicação trabalho outra entrada na forma de um reconhecedor de voz, e outras duas saídas, sendo uma um sintetizador de texto em áudio falado, e um reprodutor de ícones de áudio. Estes últimos três serviços listados foram implementados seguindo o padrão de design *Singleton*. Este padrão é útil para serviços que apresentam um único servidor, tendo a demanda de seus consumidores, ou que precisam manter um estado único para toda a aplicação (STENCEL; WĘGRZYNOWICZ, 2008).

5.5.2.1 Reconhecedor de voz

O reconhecedor de voz é implementado pela classe *VCSSpeechRecognizer*. Seu papel é, ao receber um comando de início através da chamada do seu método *startRecording()*, iniciar a captura de áudio, e reconhecimento em tempo real de fala capturada. E ao receber uma chamada do método *stopRecording()*, finalizar a captura de áudio. Como receber o texto parcialmente reconhecido é do interesse do modelo de contexto que que inicializou a captura, este deve oferecer ao serviço de reconhecimento de fala uma maneira deste avisá-lo a cada mudança do texto reconhecido. Para isto utilizou-se o atributo *delegate*. Este atributo não possui um tipo específico, requerendo apenas que a instância

a ele atribuída respeite um determinado protocolo. Neste para este caso desenvolveu-se um protocolo contendo apenas 2 métodos, sendo um para avisar a instância delegada de que há texto novo reconhecido, e o outro para avisar que encerrou-se a captura de texto, e apresentar o texto reconhecido final. Dessa forma, para utilizar o serviço de reconhecimento, o modelo de contexto deve estar em conformidade com o protocolo da instância delegada, e implementar os dois métodos de atualização de texto parcial, e fim da captura. E logo antes de iniciar as requisições ao serviço, o modelo de contexto deve inserir a si mesmo no atributo *delegate*, tornando-se a instância delegada, para então pedir o início da captura de voz. Isto pois a cada atualização de texto reconhecido, o respectivo metodo do protocolo da instância delegada será chamado, tendo o texto atualizado passado como parâmetro. E é através desta chamada que a instância delegada, ou o modelo de contexto que fez a requisição, tem acesso ao conteúdo atualizado. O mesmo vale para o fim da captura de voz, quando o método de encerramento é chamado.

O serviço de reconhecimento de voz oferecido pelo sistema operacional *iOS*, consumido pela classe *VCSSpeechRecognizer*, não pode executar mais de uma requisição de captura de cada vez. Desta forma é importante que duas classes não sejam capazes de fazer requisições simultâneas ao serviço. Esta situação aliada ao fato de ser necessário manter uma única instância delegada de quem faz as requisições, são motivos o suficiente para implementar a classe na forma de um *singleton*.

5.5.2.2 O Sintetizador de voz e o Reprodutor de de ícones de áudio

Ambos os serviço de reprodução de áudio, e de sintetização de voz possuem as mesmas restrições de poder executar apenas uma requisição por vez. Restrições estas advindas dos respectivos serviços de sistema operacional sobre os quais são implementados. O sintetizador é implementado pela classe *VCSVoiceSintesizer*, e dois tipos de métodos, cada um com uma variação para serem usadas no contexto de ajuda, onde a velocidade e entonação da fala são diferenciados. Um é o método de sintetização de texto simples, no qual coloca-se o texto a ser sintetizado como parâmetro. Há também a sintetização em sequência, no qual se coloca uma sequência de textos, para que o serviço sintetize um item por vez.

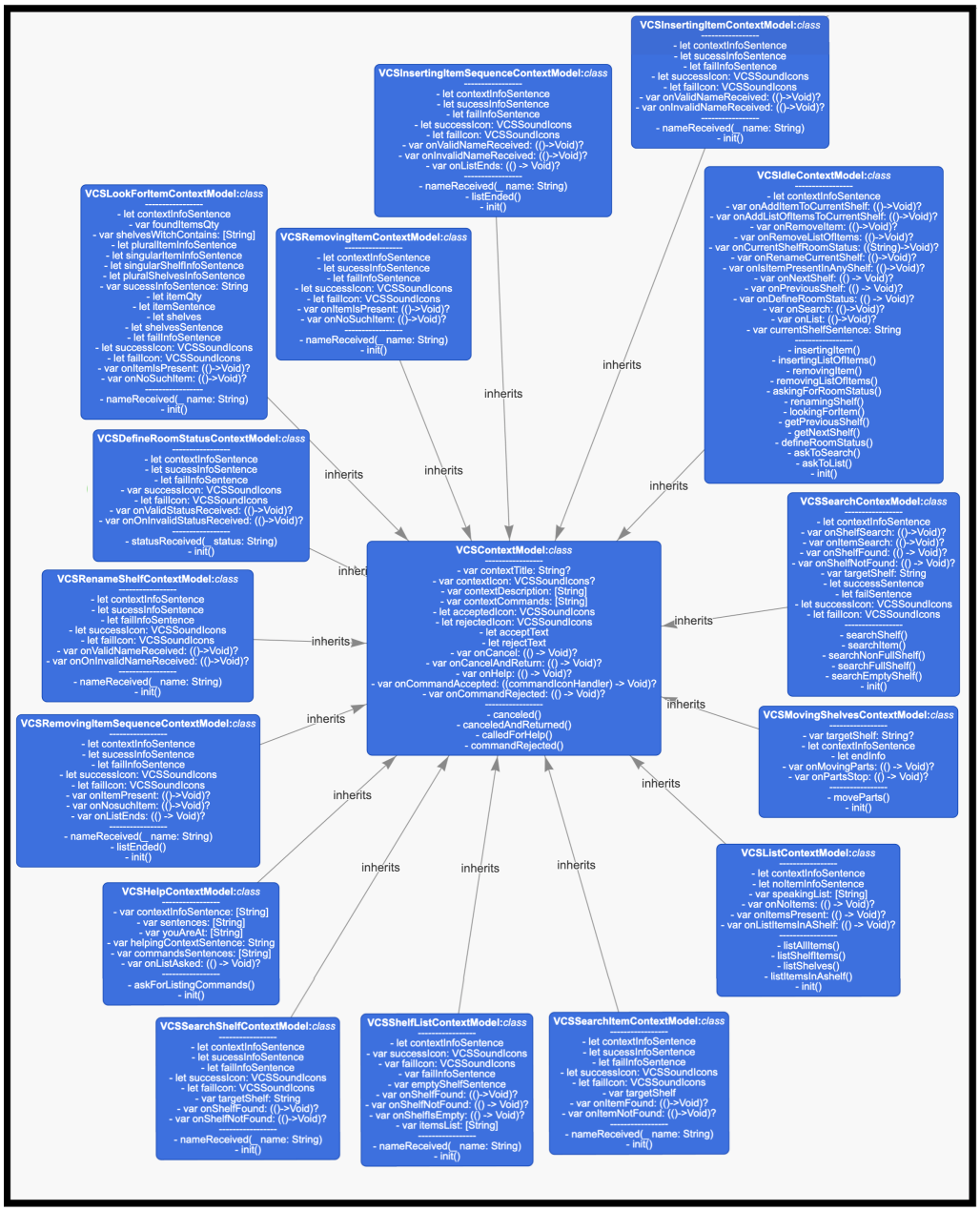
O reprodutor de ícones de áudio armazena um enumerador contendo o título de cada ícone de áudio da aplicação. Para tocar um ícone, o modelo do contexto que o requer deve chamar o método de reprodução passando o item do número de título que corresponde ao arquivo de áudio do ícone a tocar. Da mesma forma que o sintetizador e o

reconhecedor de voz, o reprodutor de áudio também é limitado a responder a uma requisição por vez. Por este motivo a classe de sintetização de voz também foi implementada como um *singleton*.

5.5.3 O gerenciador de inventário

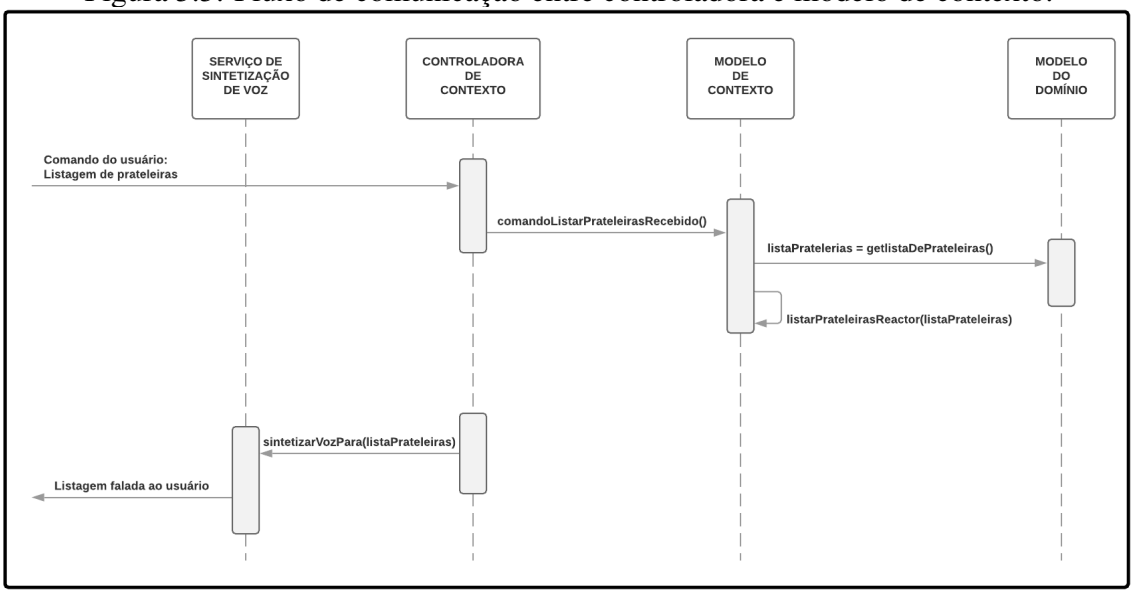
O gerenciador de inventário, por ser o responsável por manter o estado de todo inventário da aplicação, incluindo o posicionamento das prateleiras, com seus objetos, e por este motivo optou-se por implementá-lo como um *singleton*. Este possui uma lista de prateleiras, que são classes, as quais contém uma identificação numérica imutável e sequencial entre as prateleiras, um nome que pode ser definido pelo usuário, e um dicionário, definido como um conjunto de pares chave/valor. O dicionário representa os itens contidos na prateleira, sendo que a chave representa o nome o item, e o valor a sua quantidade na prateleira. No gerenciar estão implementados os métodos de inserção e remoção de itens, renomeação de prateleiras, buscas e listagens diversas. E é o gerenciador que guarda o nome da prateleira atual, sendo responsável por atualizar este valor a cada movimentação de prateleiras.

Figura 5.4: Diagrama das classes dos modelos de visualização.



Fonte: O autor, 2019

Figura 5.5: Fluxo de comunicação entre controladora e modelo de contexto.



Fonte: O autor, 2019

6 METODOLOGIA

Neste capítulo descreveremos o processo de desenvolvimento, testes e avaliação da interface de voz objeto deste trabalho. A implementação constituiu-se de duas etapas, intercaladas por uma fase prévia de testes e recolhimento de *feedbacks*.

A primeira etapa teve o objetivo de apresentar uma aplicação funcional o suficiente para que os usuários pudessem compreendê-la e serem capazes de sugerir variações textuais para os comandos implementados.

A segunda etapa de implementação destinou-se a implementar as funcionalidades, assim como endereçar as sugestões feitas pelos usuários nos testes.

6.1 Primeira etapa de desenvolvimento

A primeira etapa de desenvolvimento da aplicação manteve o focou em pontos específicos da implementação, afim de gerar uma prova de conceito de que as decisões tomadas no projeto gerariam uma aplicação funcional. Buscou-se criar uma máquina de estado que abrigasse todo o fluxo de navegação. Nesta etapa definiram-se também os comandos aceitos em cada contexto, bem como o texto que os definiria. Implementaram-se os serviços de sintetização de voz, de transcrição de voz em texto, de execução de ícones de áudio, e uma versão primária do gerenciador de inventário, não funcional, apenas com dados adicionados em código, para o uso na primeira fase de testes. Para possibilitar os testes, uma interface gráfica simplificada também foi implementada.

6.1.1 Uma máquina de estados

Inicialmente implementou-se uma máquina de estados baseada em enumeradores para abrigar toda a lógica de navegação, bem como os comandos aceitos a cada etapa do fluxo de navegação. Nela ficavam também o conteúdo textual que identificava cada um dos comandos. Em uma variável mantinha-se o estado atual da aplicação, que era representado pelo contexto da aplicação em que o usuário encontra-se a cada momento. As controladoras então tratariam de capturar os comandos de voz do usuário, através do serviço de transcrição de voz em texto, e passariam o texto a esta grande máquina de estados. Baseando-se no contexto atual da navegação e no comando recebido, a máquina de

estados dispararia as ações às controladoras e modelos de visualização, para que atuassem de acordo com estes comandos.

Esta implementação pareceu promissora, até o momento em que os comandos de um mesmo contexto passaram a requerer informações adicionais do usuário, como na inserção de um objeto, quando depois de dar o comando de inserção à interface, o usuário é chamado a dar um nome ao objeto inserido. Para endereçar estes novos estados da navegação, como a espera por um nome após o reconhecimento de um comando, novos contextos foram adicionados. Com isso os estados começaram a multiplicar, e a complexidade da máquina de estados aumentou rapidamente, tornando impraticável a manutenção e o acréscimo de mais contextos, pois todos eram gerenciados ao mesmo tempo, em um mesmo documento, assim como toda informação textual atrelada. Ainda nesta primeira fase de desenvolvimento também foi definido o conteúdo textual de cada comando. Mas a implementação já foi direcionada a aceitar variações de sentenças para um mesmo comando, e algumas poucas variações foram adicionadas. Observou-se também que a complexidade da máquina de estados que gerenciaria a navegação entre os contextos, aliada a um número, mesmo que pequeno, de variações de texto para cada comando, passou a tomar um tempo sensível para fazer o casamento entre texto reconhecido pelo reconhecedor de voz, e o conteúdo textual de cada comando.

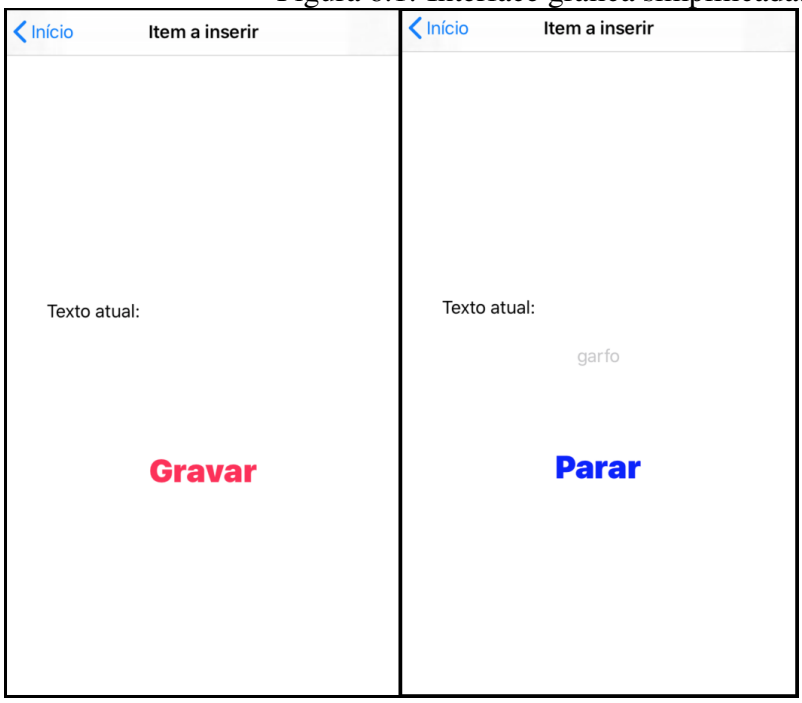
6.1.2 Serviço de voz, áudio e gerência de inventário

Os serviços de transcrição de voz em texto, de síntese de voz, e execução de ícones de áudio, implementados nesta primeira fase, não sofrendo maiores alterações na fase seguinte. Já o serviço de gerenciamento do conteúdo da prateleira viria a ser implementado apenas mais tarde. Deste modo, para a fase de testes que se seguiria, uma série de itens foi pré definida em estruturas dentro de uma classe estática, para que o conteúdo fosse consumido durante os teste. Os ícones de áudio a serem tocados a cada momento também foram selecionados e incluídos na aplicação.

6.1.2.1 Interface gráfica

Uma interface gráfica simples foi implementada para apresentar o nome do contexto atual em que o usuário se encontra, e para suportar o botão de interação. Como mostra a figura 6.1.2 à esquerda, a interface sem texto reconhecido, com o botão aguar-

Figura 6.1: Interface gráfica simplificada.



Fonte: O autor, 2019

dando o toque para gravar. E à direito este botão quando acionado, iniciando a recepção de áudios por parte da interface, que imediatamente começa a transcrevê-los em texto. Ao ser acionado novamente, encerra a recepção de conteúdo. A todo instante durante a transcrição do áudio capturado, o texto resultante vai sendo apresentado ao usuário na interface gráfica, como é visto na imagem 6.1.2 à direita.

6.2 Primeira fase de testes

A primeira fase de testes com usuários foi planejada com dois objetivos. O primeiro foi reunir mais variações de conteúdo textual para os comandos. O segundo foi estressar o fluxo de navegação, afim de encontrar inconsistências na execução da aplicação, ou problemas severos na operação da interface.

Esta primeira fase de testes não teve objetivo de avaliar a interface em si, mas buscar sugestões para sua melhoria. No entanto procurou-se seguir algumas indicações propostas em (NIELSEN, 1995), como o número de indivíduos, e a dupla execução da sequência de passos do roteiro. O número de usuários foi para dar diversidade de pontos de vista às sugestões. Já a repetição do teste como um todo serve para que o usuário possa voltar sua atenção a pontos diferentes da interface, podendo ter observações mais precisas sobre um maios número deles.

A aplicação foi testada por 7 usuários, divididos entre ambos os sexos, com idades entre 21 e 70 anos. Os sujeitos foram acompanhados pelo autor durante os testes, sendo encorajados com antecedência as suas impressões verbalmente ao autor, para que não precisassem interromper o teste para transcrever suas respostas, mas apenas após a segunda rodada de execução, pois considerou-se a primeira rodada como fase de familiaridade ao funcionamento da interface. Neste testes os usuários foram informados da existência de um contexto de ajuda, que estaria disponível durante toda a execução da aplicação. Então eram instruídos a seguir um roteiro de interação com a interface, como se estivesse controlando uma estante real. Neste roteiro os usuários passavam pelas principais funcionalidades da aplicação, inserindo comandos em cada um dos principais contextos: ajude, inicial, buscas e listagens. O roteiro foi apresentado na forma de um formulário utilizando o *Google Froms*. Este possuía apenas um campo chamado **Sugestões de comandos**. O ultimo passo do roteiro pedia para o usuário retornar ao início, e executar os passos todos uma segunda vez. O roteiro foi descrito como segue:

O aplicativo que você está prestes a testar é uma interface de voz desenvolvida para operar uma prateleira automática. Isso quer dizer que você poderá dar comandos para o celular na sua mão, e este literalmente irá entender você. Parece muito legal, mas ainda temos algumas limitações. Mesmo assim mantenha o entusiasmo, pois este teste vai servir justamente para melhorarmos esta situação.

Primeiramente não possuímos uma prateleira física para ser operada, então o teste ocorrerá apenas verbalmente. E em segundo lugar, como mencionado temos algumas limitações de comunicação, então além de ***paciência****, pedimos que se o dispositivo rejeitar algum comando seu, mesmo tendo entendido corretamente o que você falou, por favor, utilize o campo sugestão de comandos mais abaixo para nos informar a frase rejeitada, e o comando pretendido com ela. Com isto poderemos melhorar o aplicativo para entender melhor seus usuários.*

Antes de começarmos a utilizar a interface, aqui vão 3 dicas: - Você pode ***cancelar*** *qualquer ação, e* ***retornar ao início*** *a qualquer momento, bastando pedir ao aplicativo para fazê-lo. - Se precisar de* ***ajuda****, pedir à interface. Ela poderá te ajudar. - É importante para o teste que você só avance para o passo seguinte ao completar o passo em que está. Tente não pular os passos.*

Vamos começar!

- Como você pode perceber, a tela inicial do aplicativo é bem simples. ela possui este botão grande escrito ***"Gravar"****. Tente tocar nele.*

*- Este som que você ouviu significa que o aplicativo está agora escutando você. Mas ele só passa a escuto após o fim deste som. Tente pedir **ajuda** ao aplicativo.*

*- Neste momento você escutou um som, e em seguida uma segunda voz, diferente da primeira que você escutou, lhe passou uma série de informações. Esta voz será responsável por ajudar você. Sempre que a ouvir significa que está dentro do menu de ajuda. Agora peça pela **lista de comandos**.*

*- Ao fim da lista de comandos você ouviu um som, e foi informado que se encontra agora no menu **principal**. Este som significa exatamente isto, o fato de você se encontrar novamente no menu principal. A cada menu que você entrar, um som específico do menu será tocado. Tente familiarizar-se com estes.*

*- Agora vamos começar a utilizar nossa estante virtual de prateleiras. Que tal guardarmos alguns objetos virtuais nela? Vamos começar por esta **camiseta azul** virtual que se encontra ao lado da estante. De o comando de **inserção de objeto**.*

*- A estante agora está curiosa pelo objeto que você guardar. Diga a ela o nome do objeto, **"camiseta azul"**, e o coloque a camiseta na prateleira que está disponível a você neste momento.*

*- Agora que sabemos que esta prateleira irá guardar camisetas, que tal colocarmos um nome mais significativo nela? Tente **renomear a prateleira atual** com o nome **"camisetas"**.*

*- Agora temos uma prateleira para camisetas, o que significa que temos lugar para esta pilha de camisetas coloridas que alguém dobrou e deixou aqui para guardarmos. Vamos guardá-las em **sequência**, uma após a outra. Avise a prateleira de que você irá **inserir uma sequência de objetos**.*

*- Olhando para a pilha, temos 5 camisetas: uma vermelha, uma preta, uma azul, uma listrada e uma cinza. Neste momento a prateleira já espera espera o nome do primeiro objeto que você quer inserir, mas vamos com calma. Tente pedir **ajuda** para descobrir como proceder a inserção das camisetas. - O comando de inserção de sequências serve para acelerar o processo de inserção de vários itens na mesma prateleira. Agora que já o conhece, insira cada uma das 5 camisetas, sem esquecer de informar o fim da lista ao terminar:*

- ***camiseta vermelha***;
- ***camiseta preta***;
- ***camiseta azul***;
- ***camiseta listrada***;

- ***camiseta cinza***;

- Olhando bem para a essa prateleira de camisetas, acho que não cabem mais camisetas nelas. Talvez se apertarmos... mas não queremos amassar para ter que passá-las depois. Neste caso seria interessante informar a estante que a prateleira atual ***não possui mais espaço****. Tente* ***informar o novo status de espaço da prateleira****.*

- Camisetas guardadas, e tudo informado à prateleira, mas ainda temos roupas a guarda. Desta vez é esta pilha de 3 ***calças de moleton****. Como a prateleira atual está cheia, vamos* ***buscar por uma prateleira com espaço*** *para guardá-las. Primeiramente navegue até o menu* ***buscar****.*

- No menu de buscar, procure por uma ***prateleira com espaço****.*

- Como apenas uma prateleira estava cheia, nossa estante virtual apenas trouxe a prateleira seguinte até você, e esta encontra-se vazia. ***Adicione*** *estas três* ***calças de moleton*** *idênticas,* ***sequencialmente****, dando a cada uma o nome de "calça de moleton".*

- Acabamos de ser questionados se temos na prateleira uma ***camiseta xadrez****. Como é uma dúvida simples, não sendo uma busca por um item, ou a listagem de conteúdos, este comando se encontra no menu* ***principal****. Vá até ele utilizando o comando de* ***voltar ao início****, e questione sobre a* ***presença*** *da* ***camiseta xadrez*** *na estante.*

- Pois é, também esperávamos essa resposta, pois não lembrávamos de termos colocado esta camiseta na estante. Mas o que mesmo já colocamos nas prateleiras? Tente ***listar todo o conteúdo*** *da nossa estante. Comandos de* ***listagem*** *encontram-se no menu de* ***listagens****.*

- Um amigo pediu para alcançarmos uma ***camiseta azul****, e uma* ***calça de moleton*** *para ele.* ***Busque*** *e* ***remova*** *a camiseta, e em seguida a calça.*

- Agora que tiramos um camiseta da prateleira de camisetas, seria interessante ***informar*** *a estante de que* ***há espaço vago nesta prateleira****. Como estamos na prateleira das calças de moleton, que é a prateleira seguinte em relação a das camisetas, peça à estante pela* ***prateleira anterior****. E em seguida* ***informe à estante de que há espaço na prateleira****.*

O que achou da experiência? Utilize o campo dicas para dar dicas de melhorias. E se esta foi sua primeira execução do teste, pedimos a gentileza de encerrar o aplicativo, e abrir este novamente, para então repetir uma vez o teste como um todo.

6.2.1 Primeiras impressões

Esta primeira fase de testes foi feita de maneira informal, com intuito unicamente de guiar a etapa final de desenvolvimento da interface. Destacamos abaixo as impressões sobre a observação dos testes, e as observações e sugestões feitas pelos usuários.

- **Dificuldade de sincronização:** Observou-se que usuários mais velhos apresentaram mais dificuldade em sincronizar o início da fala de comando à interface para após o acionamento do botão de gravação. Com isto a comunicação do comando falhava, pois a interface não interpretava o início das palavras de comando. Uma hipótese para a ocorrência deste problema é a reprodução prolongada do ícone de áudio correspondente ao início da captura de voz para o reconhecimento de comandos. O ícone de áudio utilizado reduzia seu volume ao fim da reprodução, e os usuário, principalmente durante a primeira rodada de execução do roteiro, tenderam a tentar adiantar o fim desta reprodução falando antes deste concluir e a interface iniciar a capturas de sua vozes.
- **Menu de ajuda:** Todos usuários acessaram o menu de ajuda, e consideraram este de grande utilidade ao longo da operação da aplicação. O menu foi acionado repetidamente durante as duas repetições da execução do roteiro. No início das primeiras repetições dos passos de teste os usuários tentaram falar comandos que foram jeitados mais vezes até desistir e pedir a ajuda. Após alguns passos e alguns comandos errados, os usuários passaram a pedir a listagem de comandos mais rapidamente, ao não saberem como elaborar o comando de voz pedido. Isto pode ter tirado a oportunidade de se capturar outras variações para os comandos.
- **Demora para confirmar o comando:** O pequeno tempo de processamento da interface, gasto na tentativa de casar o texto recebido a um comando válido em determinado contexto levou os usuários a terem dúvida se a interface não avia trancado o processamento, ou entrado em um laço infinito.
- **Sugestões de variações de comandos:** Os usuários sugeriram diversas variações de comandos ao longo dos testes. Algumas variações visaram encurtar caminhos de navegação entre contextos, como ao tentar incluir com uma única frase, juntando o nome do objeto ao fim do comando de inserção, numa única frase.

Com base nas impressões obtidas nos testes primários, através das colocações por parte dos usuários, e pela observação da aplicação em funcionamento, duas decisões

principais foram tomadas. A primeira foi a de aumentar drasticamente no número de variações de frases para cada comando. A segunda foi em relação a abandonar o modelo de máquina de estados inicialmente adotado. Com o aumento das variações textuais, o tempo de processamento para verificar se determinado comando é válido ou inválido aumentou muito. E ainda a complexidade de gerenciar os contextos.

Às variações de comandos sugeridas juntamos comandos dados pelo usuário que foram rejeitados, mas não foram anotados por este no campo especificado, sendo incluído nas anotações do autor apenas. E ainda, foram adicionados comandos aceitos pela interface, mas que não eram a intenção do usuário, trazendo um funcionamento não esperado. Transcrevemos abaixo as variações sugeridas aos comandos testados.

- **Comando Ajuda:** "Me ajuda", "Me ajude"
- **Comando Listar Comandos:** "Comandos", "Lista de comandos", "Liste os comandos", "Me diz os comandos", "Listar tudo"
- **Comando de Inserção de Item:** "inserir item", "novo item", "inserir objeto", "guardar camiseta vermelha", "guardar camiseta"
- **Comando Renomear Prateleira:** "Renomear", "Trocar o nome da prateleira"
- **Comando Adicionar Itens:** "Guardar muitos objetos", "Guardar objetos", "Adicionar objetos", "Adicionar coisas", "guardar três calça de moleton"
- **Comando Fim da Lista:** "fim", "fim da sequência", "ultimo item", "fim dos objetos", "acabou"
- **Comando definir espaço na prateleira:** "prateleira cheia", "mudar espaço", "espaço da prateleira", "espaço", "prateleira com espaço", "a prateleira tem espaço"
- **Comando Buscar:** "menu buscar", "abrir menu buscar"
- **Comando Buscar Prateleira com Espaço:** "prateleira com espaço", "buscar prateleira vazia", "trazer prateleira vazia"
- **Comando Cancelar e Voltar ao Início:** "voltar", "início", "ir pro início", "reiniciar", "voltar ao início"
- **Comando Verificar Item Presente:** "tem camiseta xadrez", "camiseta xadrez", "ver item", "tem o item camiseta xadrez", "verificar item", "verificar presença"
- **Comando Listar Todos os Itens:** "Listar itens", "listar todos os itens", "Lista de tudo", "todos os itens"
- **Comando Remover Item:** "Buscar camiseta azul", "Buscar", "buscar objeto", "remover objeto"

- **Comando Prateleira Anterior:** "buscar prateleira anterior", "buscar", "ir para prateleira anterior", "trazer prateleira anterior"

6.3 Segunda etapa de desenvolvimento

Esta etapa baseou-se em desenvolver uma nova maneira de gerenciar a navegação do usuário entre os contextos da interface e de tornar mais simples o gerenciamento dos contextos em relação à manutenção da aplicação, na inclusão, remoção e alteração de contextos. Isto tornou-se imperativo com o aumento das variações de texto adicionadas a cada comando. Incluindo o gerenciamento do conteúdo de texto relacionado aos comandos contidos no contexto

6.3.1 Controladora de contextos

A solução implementada para o problema da complexidade da máquina de estados para o controle dos comandos de cada contexto, e do fluxo de navegação, foi a de distribuir a responsabilidade, antes centralizada na máquina, entre os próprios contextos. Com esta organização, a cada momento, a controladora do contexto atual teria que fazer alguns poucos casamentos de padrão de texto para descobrir se determinado comando deve ser aceito ou rejeitado, baseando-se apenas nas variações de frases que representam estes poucos comandos. Com isto o tempo de processamento para reconhecimento de comandos caiu sensivelmente. Outro efeito foi o alívio da complexidade de se fazer alterações nas rotas de navegação entre os contextos. Com a mudança, cada contexto é responsável por definir os próprios destinos de navegação, para cada comando executado, o que facilitou também a inserção de novos contextos, sem aumentar a complexidade em um componente de gerenciamento centralizado.

6.3.2 Variações textuais para os comandos de voz

As sugestões de comandos foram acrescentadas dos comandos aceitos pela interface, mas que levariam esta a uma ação que não era a intenção do usuário, e acrescentaram-se também outros comandos errôneos observados pelo autor durante os testes. Como o teste passou apenas por um subconjunto do conjunto de comandos, ficando alguns co-

mandos de fora, buscou-se avaliar as frases formuladas, e extrapolar suas variações aos demais comandos da aplicação. Com isto chegou-se ao conjunto final de comandos da aplicação apresentada a seguir.

- **Comando Cancelar:** "parar", "voltar", "retornar"
- **Comando Cancelar e voltar ao início:** "voltar ao início", "início"
- **Comando Ajuda:** "me ajuda", "ajuda", "duvida", "me ajude", "ajudar"
- **Comando Listar comandos:** "comandos", "lista de comandos", "liste os comandos"
- **Comando Verificar presença de item:** "ver item", "ver o item", "verificar item", "verificar presença"
- **Comando Adicionar item à prateleira atual:** "inserir e tem", "inserir objeto", "novo objeto", "novo item", "adicionar objeto", "guardar objeto", "guardar item", "guardar um objeto", "guardar um item"
- **Comando Adicionar lista de itens à prateleira atual:** "adicionar objetos", "adicionar sequência de itens", "adicionar sequência de objetos", "adicionar lista de itens", "adicionar lista de objetos", "inserir itens", "inserir objetos", "inserir sequência de itens", "inserir sequência de objetos", "inserir lista de itens", "inserir lista de objetos", "nova sequência de itens", "nova sequencia de objetos", "nova lista de itens", "nova lista de objetos"
- **Comando Fim da lista de inserção:** "fim", "acabou", "fim da lista", "último item", "último item da lista"
- **Comando Remover item da prateleira atual:** "remover objeto", "retirar item", "retirar objeto"
- **Comando Remover lista de itens da prateleira atual:** "remover objetos", "remover lista de itens", "remover lista de objetos", "remover sequência de itens", "remover sequência de objetos", "retirar objetos", "retirar lista de itens", "retirar lista de objetos", "retirar sequência de itens", "retirar sequência de objetos"
- **Fim da lista de remoção:** "fim", "acabou", "fim da lista", "último item", "último item da lista"
- **Atualizar status do espaço da prateleira atual:**
- **Status do espaço da prateleira atual:**
- **Renomear prateleira atual:** "trocar nome da prateleira", "trocar o nome da prateleira", "renomear prateleira", "renomear a prateleira", "trocar o nome da prateleira",

"trocar nome da prateleira", "trocar o nome da prateleira atual", "trocar nome da prateleira atual", "mudar o nome", "mudar o nome da prateleira, "mudar o nome da prateleira atual"

- **Próxima prateleira:** "próxima", "ir para próxima prateleira", "ir para a próxima prateleira"
- **Prateleira anterior:** "anterior", "ir para prateleira anterior", "ir para a prateleira anterior"
- **Buscas:** "ir para o contexto de busca", "ir para contexto de buscas", "ir para contexto de busca", "fazer busca", "ir para busca", "busca", "fazer buscas", "ir para buscas",
- **Listagens:** "listagem", "estar", "ir para listagem", "ir para listagens", "listagens", "ir para contexto de listagem", "ir para o contexto de listagem", "ir para contexto de listagens", "ir para o contexto de listagens"
- **Buscar prateleira pelo nome:** "prateleira", "prateleira pelo nome", "uma prateleira", "uma prateleira pelo nome", "buscar prateleira", "buscar uma prateleira", "buscar uma prateleira pelo nome"
- **Buscar objeto pelo nome:**
- **Buscar prateleira cheia:** "prateleira cheia", "uma prateleira cheia", "buscar prateleira cheia", "buscar uma prateleira cheia"
- **Buscar prateleira vazia:** "prateleira vazia", "uma prateleira vazia", "buscar prateleira vazia", "buscar uma prateleira vazia"
- **Buscar prateleira com espaço:** "prateleira não cheia", "prateleira com espaço", "uma prateleira não cheia", "uma prateleira com espaço", "buscar prateleira não cheia", "buscar prateleira com espaço", "buscar uma prateleira não cheia", "buscar uma prateleira com espaço"
- **Listar itens de uma prateleira:** "listar itens de uma prateleira", "listar objetos em uma prateleira", "listar objetos de uma prateleira", "itens de uma prateleira", "itens em uma prateleira", "objetos em uma prateleira", "objetos de uma prateleira", "listar itens de determinada prateleira", "listar objetos em determinada prateleira", "listar objetos de determinada prateleira", "itens de determinada prateleira", "itens em determinada prateleira", "objetos em determinada prateleira", "objetos de determinada prateleira"
- **Listar itens da prateleira atual:** "listar objetos da prateleira", "objetos da prate-

leira", "objetos na prateleira", "listar objetos da prateleira atual", "objetos da prateleira atual", "objetos na prateleira atual", "itens da prateleira", "itens na prateleira", "listar itens da prateleira atual", "itens da prateleira atual", "itens na prateleira atual", "prateleira atual", "prateleira"

- **Listar itens totais da estante:** "todos os itens", "todos os objetos", "listar todos os objetos", "listar tudo"
- **Listar prateleira presentes:** "lista de prateleiras", "prateleiras", "nomes de prateleiras", "lista das prateleiras", "nomes das prateleiras", "listar nomes das prateleiras", "listar os nomes das prateleiras"

6.4 Testes finais

A primeira fase de testes foi aplicada para verificar maiores problemas de usabilidade da interface, além de reunir sugestões de variações textuais para os comandos disponibilizados para os usuários. Em contrapartida, a fase de testes finais tem intenção de endereçar o objetivo final do trabalho. Nesta rodada final de testes os usuários avaliaram as boas práticas selecionadas para serem implementadas, que foram apresentadas na seção 5.2.

Para avaliação das boa práticas implementadas, utilizou-se um roteiro de testes similar ao utilizado para os testes primários, apresentado em 6.2, com algumas pequenas modificações. Primeiramente não foram pedidas contribuições aos usuários para apontarem melhorias na interface, mas explicitamente para avaliarem os pontos citados em 5.2. Ao fim foi adicionada uma instrução para que o usuário repita o teste três vezes, sendo que na segunda execução do roteiro, antes de iniciar, este deve alterar a experiência do usuário na aplicação, de iniciante para intermediário. E no fim, a terceira execução deve ser feita no modo experiente. Por fim os usuários foram convidados a responder um formulário, onde para cada item de boas práticas listado, este deveria dar uma nota sobre sua relevância e utilidade na interação com a interface. Após a nota de relevância foi deixado um campo de texto para o usuário deixar observações em relação ao quesito. Os testes finais foram executados por 8 usuários entre 23 e 35 anos, de ambos os sexos. Destes, 4 trabalham com tecnologia da informação, tem entre 28 e 35 anos ,sendo 3 homens e uma mulher. Os outros 4 estão na faixa etária entre 23 e 25 anos, dendo duas mulheres e dois homens com ocupações não relacionadas à tecnologia da informação, ou qualquer ramo

relacionado à computação.

Os passos para a execução da aplicação nos testes finais foram os mesmo dos testes iniciais. Logo após a identificação destes passos foi colocado o formulário para qualificar a relevância dos quesitos de teste. Novamente foi utilizado o serviço *Google Forms* para a produção do formulário, e este foi apresentado aos usuários como segue:

Responda às questões abaixo apenas após concluir as três rodadas de teste do roteiro acima.

Se você já concluiu as três rodadas do teste, agora é o momento de avaliar alguns quesitos dessa interface. Os quesitos estarão descritos, mas se houver dúvidas sobre o que se tratam na aplicação, sinta-se à vontade para perguntar.

Para cada quesito, primeiramente dê a este um conceito de 1 a 5, sendo 1 se o quesito atrapalhou muito a utilização do aplicativo, 3 se o quesito foi irrelevante para a utilização, e 5 se o quesito foi bem empregado facilitando o uso do aplicativo. Após dar o conceito ao quesito, você pode adicionar observações no campo de observações logo abaixo deste.

Os quesitos:

Clareza dos diálogos: *A voz utilizada pelo aplicativo para falar com o usuário foi gerada sinteticamente. Para evitar mal entendimentos, procurou-se usar linguagem simples, e frases que não dessem margem à interpretação. Procurou-se apenas informar o usuário, sem fazer-lhe perguntas. Qual conceito, de 1 a 5, você dá a este quesito?*

Número de comandos reduzidos: *Como você pode perceber, cada comando tinha um menu correto para estar, com exceção do menu de ajuda, ou dos comandos de voltar e cancelar. Esta organização foi aplicada para que o usuário tenha poucos comandos a decorar a cada momento, e sabendo o menu em que ele se encontra, fique mais fácil de saber o que pedir à prateleira. Qual conceito, de 1 a 5, você dá a este quesito?*

Vozes: *Foram usadas duas vozes diferentes no aplicativo. Uma para dar informações dentro dos menus, e outra levemente mais grave e mais lenta, para falar durante os momentos de ajuda. O objetivo foi deixar o usuário a par de que estava em um menu de ajuda.*

Sons: *Além das falas da interface para dar informações ao usuário, Ícones de áudio (sons não falados) foram tocados ao longo de todo o uso do aplicativo. Estes sons tinha o objetivo de informar o usuário sobre tudo o que acontecia, como abertura de menus, comandos aceitos ou rejeitados, ou como a conclusão de tarefas.*

Adaptação à expertise do usuário: *Como você pode ter percebido, a cada rodada*

de execução do roteiro de teste, a interface passava a falar menos, pois a cada rodada aumentávamos a expertise do usuário. Esta possibilidade foi desenvolvida para permitir que à medida que esta pessoa se familiarize com a interface, ela possa tirar do caminho áudios que não lhes são mais interessantes, sem perderem a referência do que está acontecendo. Qual conceito, de 1 a 5, você dá a este quesito?

Interface gráfica com texto reconhecido: *O aplicativo possui uma tela bastante simplificada. Procurou-se colocar nela apenas o essencial. E um dos itens foi a transcrição em tempo real das falas do usuário reconhecidas pelo aplicativo. O intuito foi manter o usuário informado do que está sendo entendido daquilo que ele fala, para que ele perceba qualquer erro de interpretação. Qual conceito, de 1 a 5, você dá a este quesito?*

Avisos: *A interface avisa constantemente sobre tudo o que acontece, como se um comando foi aceito ou rejeitado, ou se determinado nome não é possível de ser utilizado para um objeto. Qual conceito, de 1 a 5, você dá a este quesito?*

Menu Global: *Os comandos de ajuda, cancelar e voltar foram deixados como acessíveis a qualquer momento na aplicação. Isto foi feito por se julgar importante o acesso constante a estes comandos, mas também para dar a impressão de que o usuário está sempre dentro de um menu global. Qual conceito, de 1 a 5, você dá a este quesito?*

Menu de ajuda: *O menu de ajuda foi colocado na aplicação para que o usuário não precise parar o que está fazendo, para ir a outro lugar no aplicativo para ler instruções do que pode ser feito. O objetivo foi o acesso ser o mais direcionado o possível para o momento de necessidade na aplicação. Qual conceito, de 1 a 5, você dá a este quesito?*

7 RESULTADOS

7.1 Testes primários

Durante a observação das rodadas de teste finais foi possível perceber que os usuários acabaram utilizando algumas variações de sentenças sugeridas pelos usuários dos testes primários. Mas ainda assim a grande maioria das frases elaboradas se aproximavam principalmente das palavras utilizadas para descrever o comando nos passos do roteiro. Por exemplo, ao pedir para o usuário dar o comando de inserção de objeto, as variações tenderam a utilizar ou a palavra inserção, ou a palavra objeto, em detrimento de palavra como adicionar, ou item.

7.2 Testes finais

Os conceitos sobre cada quesito, obtidos através do formulário no *Google Forms*, estão aqui representados graficamente.

Antes do início dos testes, encorajou-se os usuários a deixarem observações para os quesitos sobre os quais foram questionadas a relevância no uso. Com isto foram deixadas ao todo 20 textos de observação. Estes serão transcritos à seguir.

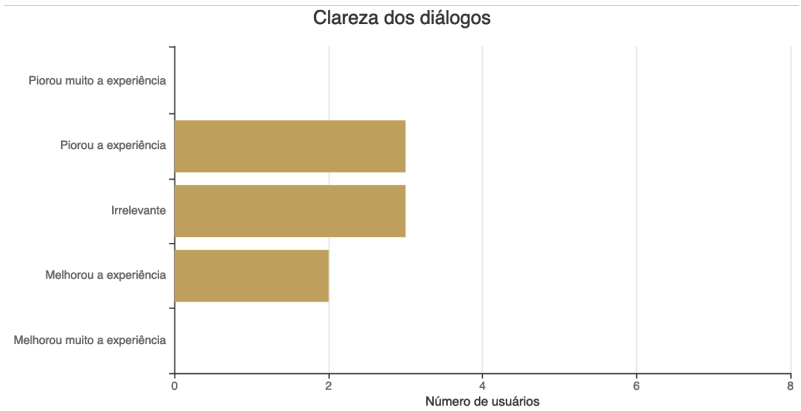
7.2.1 Clareza dos diálogos

"As falas são boas, mas a voz é estranha."

"Parece que você fala com um robô."

Vemos a impressão dos usuários sobre a clareza dos diálogos na figura 7.2.1. A distribuição se encontra ao redor da irrelevância, com algumas pessoas vendo alguma vantagem. O objetivo foi o de avaliar a simplicidade das falas da máquina, mas olhando as duas observações às quais tivemos acesso, estas mostram descontentamento com a voz utilizada me si. Talvez seja a voz sintetizada a causa da rejeição neste quesito. Alguma pessoas consideraram como algo que melhorou a experiência, mas não escreveram o por quê.

Figura 7.1: Impressão dos usuários em relação à clareza dos diálogos.

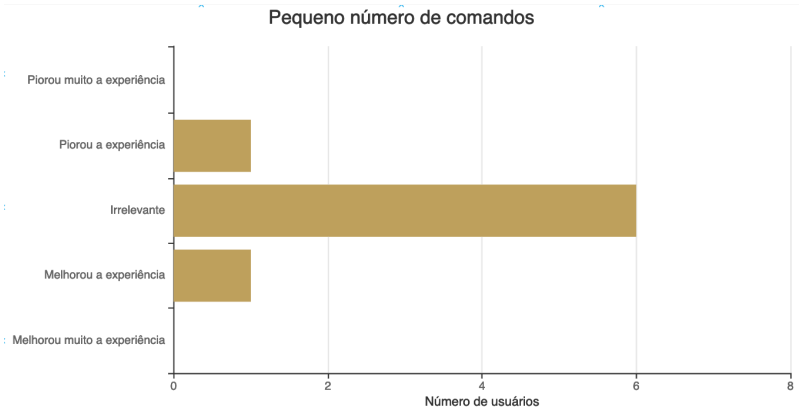


Fonte: O autor, 2019

7.2.2 Número de comandos reduzidos

"Ficou ruim ter que ficar indo e voltando. Só que organizado é melhor. Poderia ser mais rápido."

Figura 7.2: Impressão dos usuários em relação à organização dos comandos.



de comandos.png Fonte: O autor, 2019

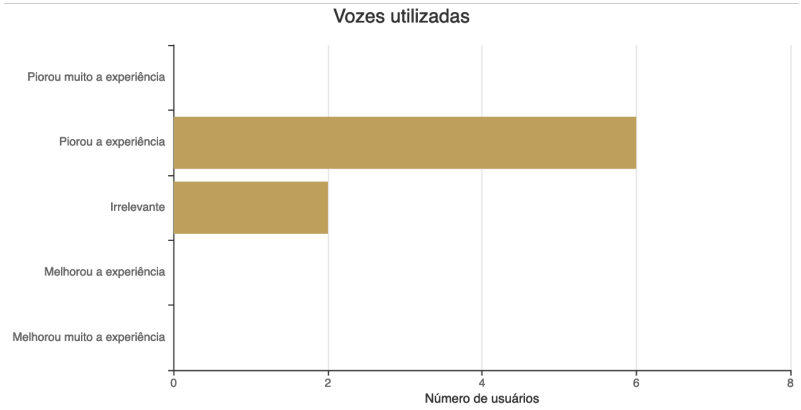
Na figura 7.2.2, que representa a impressão dos usuários em relação à distribuição dos comandos no contexto, vemos umas distribuição das impressões dos usuários mais próxima da irrelevância. Se olharmos a observação que obtivemos de um usuário, a distribuição dos comandos em contextos é sim vantajosa. No entanto talvez não tenhamos aplicado a melhor configuração destes. Algum outro arranjo de comandos e contextos pode tornar mais eficiente a navegação.

7.2.3 Vozes

"A outra voz ficou ruim."

"Ter outra voz não melhorou, até porque ela falava quando tu entrava na ajuda."

Figura 7.3: Impressão dos usuários em relação ao uso de vozes diferentes.



Fonte: O autor, 2019

Neste quesito buscou-se avaliar a relevância do uso de uma voz diferente para o contexto de ajuda. A segunda voz falava levemente mais devagar e num tom um pouco mais grave, para deixar evidente o contexto de ajuda. Como vemos na figura 7.2.3, na sua maioria, os usuários rejeitaram o uso da segunda voz. Nas duas observações que obtivemos vemos uma tratando de não ver a necessidade desta mudança de voz, e outra tratando de como a voz ficou artificial.

7.2.4 Ícones de áudio

"Uns sons estavam mais altos que os outros. Ainda assim achei bem legal. O de gravar era muito barulhento."

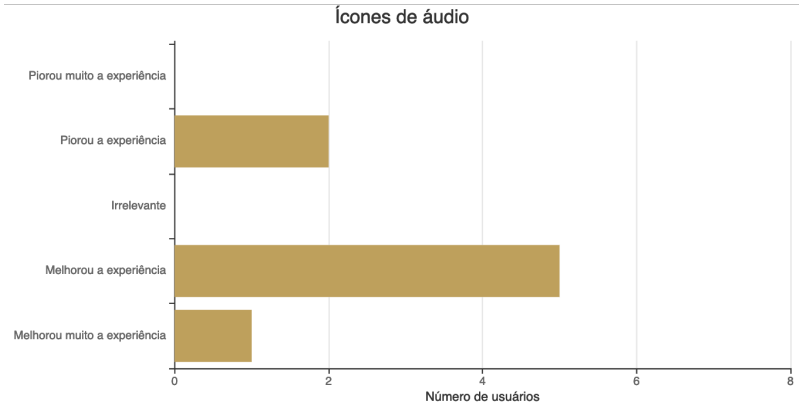
"O som do início repetia muito, e era agudo e alto."

"O som de que aceitou e o de que não aceitou podiam tocar mais curtinhos."

"Gostei do barulho das prateleiras hehe"

Neste quesito procurou-se avaliar a impressão dos usuários sobre os ícones de áudio reproduzidos em resposta às suas ações e às da interface. Na figura 7.2.4 vemos uma clara aceitação da maioria dos usuários. Alguns julgaram que a presença dos ícones prejudicou a experiência, o que faz sentido quando vemos as observações deixadas. Estas foram em relação à qualidade dos áudios em si, e não em relação à presença destes.

Figura 7.4: Impressão dos usuários em relação ao uso de ícones de áudio.



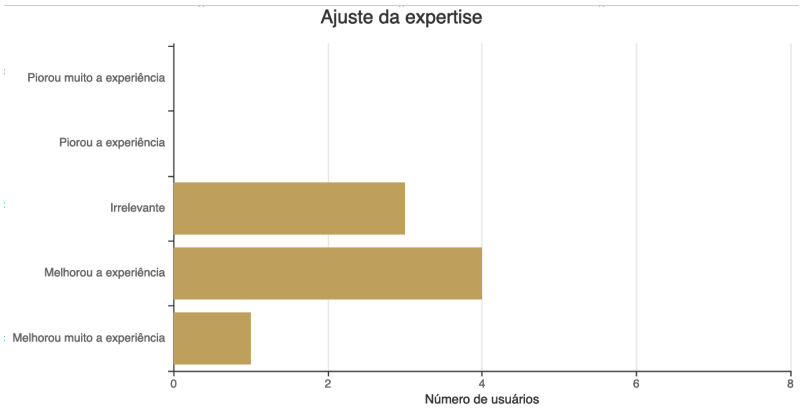
Fonte: O autor, 2019

Alguma melhoria da qualidade destes poderia ser suficiente para corrigir isto.

7.2.5 Adaptação à expertise do usuário

> *"Achei difícil sem as falas no fim, mas tirar umas partes deixou mais rápido."*
>
> *"Depois de mais um tempo treinando daria pra usar só com os áudios."*

Figura 7.5: Impressão dos usuários em relação à possibilidade de se ajustar à expertise dos usuários.



Fonte: O autor, 2019

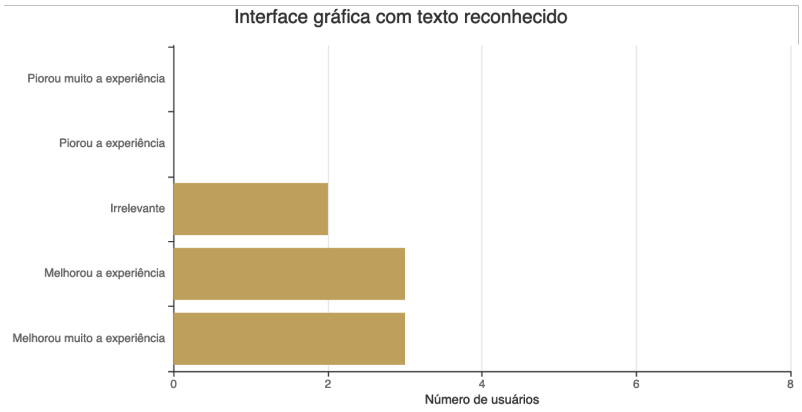
Neste quesito avaliou-se a possibilidade dos usuários ajustarem a quantidade de falas da interface à sua expertise no uso desta. Como vemos na figura 7.2.5, na sua maioria os usuários viram como positiva esta possibilidade. Na observações os usuários apontaram a possibilidade de se guiar utilizando apenas os ícones de áudio, quando se tiradas as falas da máquina, e de a interface se tornar mais rápido. Isto devido ao tempo de espera até que a interface conclua suas falar, para que então o usuário possa iniciar a

sua fala.

7.2.6 Interface gráfica com texto reconhecido

"Legal poder ficar olhando, mas quando errava não dava pra cancelar."
"Tinha que cuidar pra falar devagar senão não pegava."

Figura 7.6: Impressão dos usuários em relação à interface gráfica.



Fonte: O autor, 2019

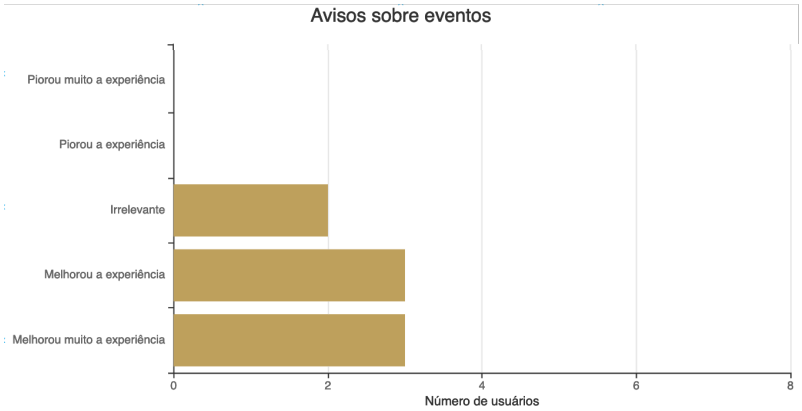
Neste quesito buscou-se avaliar principalmente a impressão dos usuários à transcrição imediata do texto reconhecido de suas falas pela interface de voz. Na figura 7.2.6 vemos claramente que os usuários consideraram bastante vantajosa a presença deste comportamento. As observações obtidas foram negativas, mas não tiveram relação com a presença do texto escrito em si, e sim sobre a impossibilidade de cancelar um comando ao perceber que este foi mal interpretado, ou de problemas do reconhecedor de voz.

7.2.7 Avisos

"Muito avisos"
"Com o som dava pra saber que aceitou o comando, não precisava avisar sempre."

Neste quesito buscou-se avaliar a impressão do usuário sobre os avisos espontâneos da máquina, que eram disparados em resposta a eventos da máquina, sem serem diretamente requisitados pelo usuário. Na figura 7.2.7 vemos uma grande aceitação desta informação pelos usuários. Nas observações recebidas, vemos que talvez estes fossem muito frequentes, ou trouxessem informação irrelevante em alguns momentos. Mas a aceitação foi clara.

Figura 7.7: Impressão dos usuários em relação aos avisos disparados.



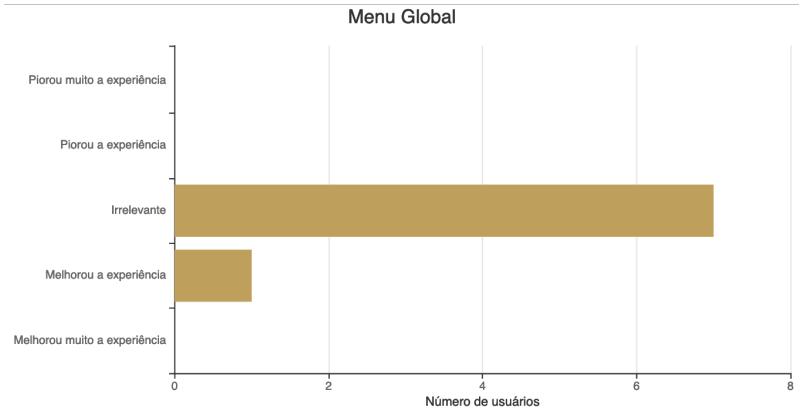
Fonte: O autor, 2019

7.2.8 Menu Global

"Não achei que era um menu"

"As vezes eu queria só voltar, e ia pro início."

Figura 7.8: Impressão dos usuários em relação ao uso de vozes diferentes.



Fonte: O autor, 2019

Neste quesito vemos a impressão do usuário em relação aos comandos globais, como cancelar, voltar e ajuda. Como podemos ver na figura 7.2.8, a presença destes comandos como globais foi vista como irrelevante, com alguma pessoas vendo vantagens na presença destes comandos globalmente.

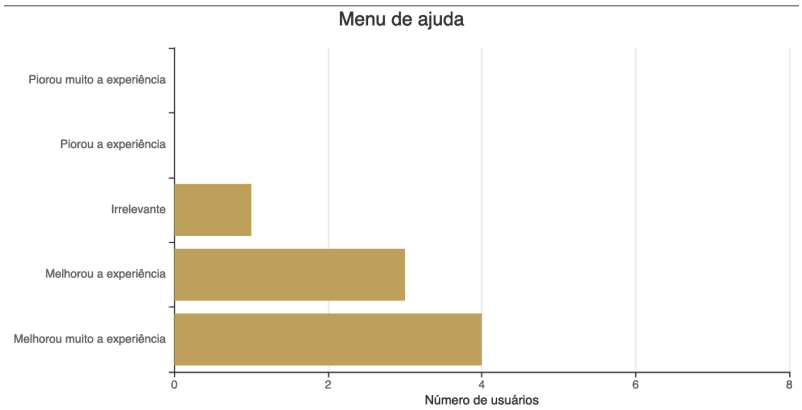
7.2.9 Menu de ajuda

"A ajuda era rápida"

"Estava fácil de usar."

"Podia dar pra arrumar a velocidade dos áudio na ajuda."

Figura 7.9: Impressão dos usuários em relação ao menu de ajuda disponibilizado.



Fonte: O autor, 2019

Entre os quesitos apresentados, o que se refere à presença de um menu de ajuda foi o mais bem aceito. Como vemos na figura 7.2.9 ficou clara a vantagem observada na presença deste menu. Nas observações observou-se também a facilidade de se chegar a este, oque se deve justamente ao fato de este ser acessado diretamente pelo menu global.

8 CONCLUSÃO

Este trabalho produziu uma interface de voz para validar com usuários a relevância de boas práticas na implementação de interfaces de voz. Para se chegar numa interface funcional, que viabiliza-se a validação com usuários, buscou-se primeiramente reunir variações textuais para os comandos, que permitissem aos usuários dar sentido aos comandos dados à interface. Propô-se uma organização dos comandos em contextos, para facilitar tanto a implementação da interface, quanto a navegação dos usuários para execução dos comandos durante o uso. Ao fim, foi proposta uma arquitetura de código desenhada a partir do uso de padrões de design já utilizados.

Como observado nos testes finais, o uso de variações textuais teve efeito, na medida que os usuários deram comandos à máquina utilizando estas variações. Sem etas o número de comandos não reconhecidos teria cido maior, o que talvez inviabilizasse a validação de outros aspectos por piorar muito a experiência dos usuários.

O uso de contexto de interação não teve muito efeito sobre os usuários. Talvez neste contexto de controle de uma estante de prateleiras, por conter um conjunto bastante coeso de comandos, não haja vantagem em organizar comandos em tantos contextos diferente. Ou ainda, uma distribuição melhor de contextos e comandos possa ter resultados melhores neste quesito.

De maneira geral as boas práticas implementadas tiveram impacto na experiência de uso dos usuários de teste. Alguns quesitos foram considerados irrelevantes, e até prejudiciais, como o uso de voz sintética para facilitar o desenvolvimento. Já a presença de um menu de ajuda, de uma interface gráfica com apresentação imediata das falas reconhecidas, e ainda, de avisos constantes tiveram uma resposta positiva por parte dos usuários. Estes quesitos positivos tem em comum o objetivo de dar mais controle sobre a aplicação ao usuário. Este podendo questionar a interface sobre o que está acontecendo a cada momento, poder observar o que esta está assimilando do que este está falando, e ainda ser avisado sobre eventos que ocorrem, permite que este possa observar com mais clareza o que acontece, e tenha que supor muito menos.

9 TRABALHOS FUTUROS

Como observado em (YANKELOVICH; LEVOW; MARX, 1995a), a utilização de áudio gravados é preferível a áudios criados com sintetizadores de voz, tanto pela familiaridade que a voz natural trás, quanto pela falta de prosódia que acompanha os textos sintetizados. Com isto uma primeira proposta de trabalho futuro é a substituição dos áudios sintetizados nas falas da interface que são fixadas na aplicação. Deixando para sintetizar apenas os áudios sintetizados a partir de nomes de prateleiras e objetos. Com isto, além de maior naturalidade nas informações faladas pela interface, a inclusão da prosódia permitirá às interfaces questionar o usuário deixando claro que esta está fazendo uma pergunta.

Outra proposta de trabalho é uma integração da interface com a *Siri*, que é a interface de voz implementada no sistema operacional *iOS*. Esta integração permitiria utilizar o poder de interpretação de linguagem natural da *Siri* para tornar o controle da estante de prateleiras muito mais fluida e natural, além de retirar a necessidade de abrir um aplicativo específico para operar a estante, bastando apenas dar os comandos diretamente ao *smartphone*.

REFERÊNCIAS

ARNOLD, G.; KLENNER, S. Acessibilidade doméstica: Projeto de estante para auxilio de idosos. **INOVAMUNDI, anais**, Feevale, 2015.

BICKENBACH, J. The world report on disability. **Disability & Society**, Taylor & Francis, v. 26, n. 5, p. 655–658, 2011.

BILMES, J. A. et al. The vocal joystick: A voice-based human-computer interface for individuals with motor impairments. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the conference on human language technology and empirical methods in natural language processing**. [S.l.], 2005. p. 995–1002.

COHEN, M. H. et al. **Voice user interface design**. [S.l.]: Addison-Wesley Professional, 2004.

COLEMAN, D. et al. Using metrics to evaluate software system maintainability. **Computer**, IEEE, v. 27, n. 8, p. 44–49, 1994.

DEWSBURY, G.; EDGE, M. Designing the home to meet the needs of tomorrow. **Open House International**, v. 26, n. 2, p. 33–42, 2001.

FOWLER, M. **Patterns of enterprise application architecture**. [S.l.]: Addison-Wesley Longman Publishing Co., Inc., 2002.

GAMMA, E. et al. Design patterns: Abstraction and reuse of object-oriented design. In: SPRINGER. **European Conference on Object-Oriented Programming**. [S.l.], 1993. p. 406–431.

HAMILL, M. et al. Development of an automated speech recognition interface for personal emergency response systems. **Journal of NeuroEngineering and Rehabilitation**, BioMed Central, v. 6, n. 1, p. 26, 2009.

IPIÑA, D. López-de; LORIDO, T.; LÓPEZ, U. Blindshopping: enabling accessible shopping for visually impaired people through mobile technologies. In: SPRINGER. **International Conference on Smart Homes and Health Telematics**. [S.l.], 2011. p. 266–270.

KAMM, C. User interfaces for voice applications. **Proceedings of the National Academy of Sciences**, National Acad Sciences, v. 92, n. 22, p. 10031–10037, 1995.

LINES, L.; HONE, K. Multiple voices, multiple choices: Older adults' evaluation of speech output to support independent living. **Gerontechnology**, v. 5, n. 2, p. 78–91, 2006.

MOIR, T. J. et al. From science fiction to science fact: a smart-house interface using speech technology and a photo-realistic avatar. In: IEEE. **2008 15th International Conference on Mechatronics and Machine Vision in Practice**. [S.l.], 2008. p. 327–333.

NIELSEN, J. How to conduct a heuristic evaluation. **Nielsen Norman Group**, v. 1, p. 1–8, 1995.

NUSSBAUM, G. People with disabilities: assistive homes and environments. In: SPRINGER. **International Conference on Computers for Handicapped Persons**. [S.l.], 2006. p. 457–460.

O'NEILL, A. et al. Initial evaluations of a robot mobility aid for frail and elderly visually impaired persons. **Assistive Technology on the Threshold of the New Millennium**, IOS Press, 1999.

PARK, I.-C.; LEE, C.-M. User interface design model for improving visual cohesion. **Journal of the Korea Academia-Industrial cooperation Society**, The Korea Academia-Industrial cooperation Society, v. 12, n. 12, p. 5849–5855, 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**, springer-verlag, v. 17, n. 1, p. 127–144, 2013.

PROKOPEC, A. Pluggable scheduling for the reactor programming model. In: **Programming with Actors**. [S.l.]: Springer, 2018. p. 125–154.

QURESHI, M.; SABIR, F. A comparison of model view controller and model view presenter. **arXiv preprint arXiv:1408.5786**, 2014.

RASHID, Z. et al. Using augmented reality and internet of things to improve accessibility of people with motor disabilities in the context of smart cities. **Future Generation Computer Systems**, Elsevier, v. 76, p. 248–261, 2017.

RIMMER, J. H.; ROWLAND, J. L. Health promotion for people with disabilities: Implications for empowering the person and promoting disability-friendly environments. **American journal of lifestyle medicine**, SAGE Publications Sage CA: Los Angeles, CA, v. 2, n. 5, p. 409–420, 2008.

ROSENFELD, R.; OLSEN, D.; RUDNICKY, A. Universal speech interfaces. **interactions**, Citeseer, v. 8, n. 6, p. 34–44, 2001.

SIXSMITH, A.; SIXSMITH, J. Ageing in place in the united kingdom. **Ageing International**, Springer, v. 32, n. 3, p. 219–235, 2008.

STENCEL, K.; WĘGRZYNOWICZ, P. Implementation variants of the singleton design pattern. In: SPRINGER. **OTM Confederated International Conferences"On the Move to Meaningful Internet Systems"**. [S.l.], 2008. p. 396–406.

STIFELMAN, L. J. et al. Voicenotes: a speech interface for a hand-held voice notetaker. In: ACM. **Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems**. [S.l.], 1993. p. 179–186.

SYROMIATNIKOV, A.; WEYNS, D. A journey through the land of model-view-design patterns. In: IEEE. **2014 IEEE/IFIP Conference on Software Architecture**. [S.l.], 2014. p. 21–30.

YANKELOVICH, N.; LEVOW, G.-A.; MARX, M. Designing speechacts: Issues in speech user interfaces. In: **CHI**. [S.l.: s.n.], 1995. v. 95, p. 369–376.

YANKELOVICH, N.; LEVOW, G.-A.; MARX, M. Designing speechacts: Issues in speech user interfaces. In: **CHI**. [S.l.: s.n.], 1995. v. 95, p. 369–376.

YASUMURA, M.; YOSHIDA, R.; YOSHIDA, M. Prototyping and evaluation of new remote controls for people with visual impairment. In: SPRINGER. **International Conference on Computers for Handicapped Persons**. [S.l.], 2006. p. 461–468.