UNIVERSIDADE DO SUL DE SANTA CATARINA

LUCAS NUNES VENTURA

LUCAS TESTONI SCHMITT

SISTEMA MÓVEL CONTROLADO POR VOZ PARA AUXILIAR NO REGISTRO DE MEDIÇÃO FÍSICA DA OBRA NO CONTEXTO DA CONSTRUÇÃO CIVIL

Florianópolis

2015

LUCAS NUNES VENTURA
LUCAS TESTONI SCHMITT

SISTEMA MÓVEL CONTROLADO POR VOZ PARA AUXILIAR NO REGISTRO DE MEDIÇÃO FÍSICA DA OBRA NO CONTEXTO DA CONSTRUÇÃO CIVIL

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação da Universidade do Sul de Santa Catarina, como requisito parcial à obtenção do título de Bacharel em Sistemas de Informação

Orientador: Prof. Saulo Popov Zambiasi, Dr.

Florianópolis

2015

LUCAS NUNES VENTURA
LUCAS TESTONI SCHMITT

SISTEMA MÓVEL CONTROLADO POR VOZ PARA AUXILIAR NO REGISTRO DE MEDIÇÃO FÍSICA DA OBRA NO CONTEXTO DA CONSTRUÇÃO CIVIL

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação da Universidade do Sul de Santa Catarina, como requisito parcial à obtenção do título de Bacharel em Sistemas de Informação

Florianópolis, 17 de novembro de 2015.

Professor e orientador Saulo Popov Zambiasi, Dr.
Universidade do Sul de Santa Catarina

Prof. Daniella Vieira, M Eng.
Universidade do Sul de Santa Catarina

Prof. Sheila Santisi Travessa, M Eng.
Universidade do Sul de Santa Catarina

AGRADECIMENTOS – Lucas Nunes Ventura

Agradeço primeiramente a minha mãe Rosane Nunes Farias, sem ela não teria condições de ter iniciado a faculdade e graças a ela, estou aqui chegando ao final de mais uma caminhada. Por sempre me apoiar e não me deixar relaxar nos momentos de necessidade e estar sempre presente nos momentos difíceis e memoráveis, por me ensinar o certo e o errado e não deixar eu me perder no caminho. Em especial a meu pai (Edimilson Ventura) falecido em 2015, por tudo que me ensinou e por ter tido um papel indispensável na formação do meu caráter, ele não teve a oportunidade de presenciar a concretização deste sonho, mas tenho certeza que a experiência passada a mim foi de extrema importância, sei que torce de onde ele está, para que eu tenha sucesso nesta nova jornada que inicia.

A minha avó (Zulmeia Simas Farias) pela ajuda financeira e pelo apoio constante na minha vida pessoal e profissional e pela disponibilização de recursos facilitadores, que tornaram essa caminhada menos custosa.

Não poderia deixar de agradecer a minha grande amiga e namorada Stéphanie da Silva Leal por estar presente em todos os momentos difíceis até aqui, por me ajudar a focar e lembrar do meu objetivo que era chegar até aqui, pela compreensão e paciência e por ser uma das principais motivações na minha vida.

Agradeço também aos meus amigos e colegas de trabalho, que me ensinaram muito nesses dois últimos anos e ter dado a base inicial para que fosse possível o desenvolvimento do software presente neste trabalho.

E por fim a Softplan por ter me dado a oportunidade profissional pelo apoio no desenvolvimento deste trabalho e pela ajuda financeira através do programa do tratado de aprendizagem, ao Luciano Cervo, responsável por este trabalho dentro da Softplan, por ter me ajudado com a ideia inicial deste tema e ter sido atencioso e disponível nos momentos de dúvidas e esclarecimentos.

AGRADECIMENTOS – Lucas Testoni Schmitt

A meu pai Ciro Júlio Schmitt e minha mãe Rosa Maria Testoni Schmitt, que me deram todo o apoio e motivação sempre nas horas precisas.

A toda minha família Testoni e Schmitt pelas orações realizadas a mim.

A todos meus colegas de trabalho e amigos, que propuseram em ajudar em qualquer situação que fosse e acima de tudo, entenderam sempre.

Ao meu Orientador e Professor Dr. Saulo Popov Zambiasi, pela paciência e dedicação de seu tempo para proporcionar uma boa elaboração deste trabalho.

A todos um muito obrigado.

“Viva uma vida boa e honrada. Assim, quando você ficar mais velho e pensar no passado, poderá obter prazer uma segunda vez”. – Dalai Lama

RESUMO

Em um canteiro de obra da área da construção civil, têm-se algumas maneiras para realizar as medições das tarefas diárias, para controle interno das atividades exercidas naquele dia. Estas maneiras vão desde a utilização de softwares ERP até planilhas para realização das inclusões das medições, porém problemas nas inclusões podem surgir, tais como o esquecimento da informação gerada, que só depois é inserida no sistema ou, outro exemplo, o esquecimento da planilha em algum local, fazendo com que perca as informações anotadas daquele dia, gerando um retrabalho. Neste contexto, pensou-se no desenvolvimento de um sistema de auxílio no registro destas medições via comando de voz em dispositivos móveis. Para isto, necessitou-se de uma pesquisa elaborada sobre a tecnologia de reconhecimento de voz e processos do canteiro de obra de uma construção civil, para que então pudesse partir para a prototipação e aplicação deste sistema e futura integração de todas as tecnologias e softwares envolvidos. Como resultado, o sistema comportou-se de maneira adequada quando em modo online, agregando agilidade no processo de registro de medições.

Palavras-chave: Protótipo, Construção civil, Reconhecimento de voz, Tecnologia

ABSTRACT

In a construction site of the construction area, there have been few ways to make measurements of daily tasks, for internal control of the activities performed that day. These ways range from the use of ERP software to spreadsheets to perform the inclusion of measurement, but problems in the inclusions may arise, such as forgetfulness of information created, which only then is entered into the system or, another example, the spreadsheet oblivion in somewhere, causing it to lose the information recorded that day, generating a rework. In this context, it was though in the development of an aid system at the record of these measurement via voice command on mobile devices. For this, needed a research carried out on voice recognition technology and the construction site of the processes of construction, so that we could build the prototyping and application of this system and future integration of all technology and software involved. As a result, the system behaved properly when in online mode, adding flexibility in measurement of the registration process.

Keywords: Prototype, Building, Speech Recognition, Technology

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE GRÁFICOS

LISTA DE CÓDIGOS

SUMÁRIO

1 INTRODUÇÃO

A construção de uma obra no campo de atuação da engenharia civil faz parte de um dos objetivos cruciais na execução de um projeto. Esta atividade é garantida pelo esforço de seres humanos e máquinas, onde a ausência ou o aperfeiçoamento de um desses elementos, implica diretamente no planejamento do projeto. A questão do aprimoramento ou evolução dos instrumentos que apoiam o trabalho do homem, proporcionam um grau de melhoramento na execução do projeto e assim, conseguindo atender as exigências dos clientes, que hoje cobram pelo aceleramento na finalização destes projetos (PEURIFOY, 2011).

Esta requisição dos clientes, faz com que a empresa executora do projeto, procure alternativas que possam atender estas demandas. Para Xavier et al (2014, p. 3), "busca a eliminação ou supressão das atividades que não agregam valor, focando no aumento da eficiência das atividades que agregam valor através do melhoramento contínuo dos processos somado à implantação de novas tecnologias". Ou seja, para um melhor aperfeiçoamento, deve-se verificar os processos, que fazem parte deste projeto e analisá-los a nível de estruturação das atividades, reduzindo, assim, gargalos existentes, não esquecendo da busca contínua de novas tecnologias, pois estas, proporcionam qualidade quando bem utilizadas.

Para Cassimiro (2013):

> [..] A tecnologia da informação apresenta-se como uma importante ferramenta à disposição dessas empresas para que consigam desempenhar seus processos e projetos de forma qualitativa, controlada e precisa. A implantação de um sistema proporciona uma série de vantagens: aumenta o nível de organização interna, o controle da administração, a produtividade, redução de custos e de erros, melhorando, assim, a credibilidade junto a seus clientes.

Segundo Farah (1988), as transformações tecnológicas no subsetor de edificações ao longo do seu desenvolvimento, não são significativas ou estão em processo lento. Exemplificando, em um registro de medição física de uma obra onde uma equipe fica responsável por medir quanto de um determinado trabalho foi executado, mesmo utilizando-se de recursos mais avançados como aplicativos em dispositivos móveis, o apontador das informações ainda precisa utilizar alguns artefatos para fazer o efetivo levantamento do quantitativo, tais como, fitas métricas, réguas, prumos, medidores por infravermelho, entre outros. Contudo, mesmo ainda tendo muitos artefatos diferentes para trazer uma variável única, cabe às empresas estarem percebendo tendências de mercado, sendo alguma

transformação tecnológica ou algo direcionado para o negócio, mercado, pois isto, proporciona uma vantagem sobre as outras empresas (LENZI et al, 2010) e, assim, com os investimentos necessários para que as transformações tecnológicas acontecem, muitos destes artefatos viram apenas um aplicativo ou um artefato.

O presente trabalho tem motivações junto a empresa Softplan, tendo como objetivo descrever a importância que a tecnologia da informação tem na área da construção civil e, com isso apresentar um aprimoramento no sistema Sienge Mobile[1] da Softplan, mais precisamente no módulo de registro de medição de obras. Como resultado, espera-se que o sistema citado, possa proporcionar agilidade na execução do mapeamento do dia a dia da obra, acessibilidade para que isto ocorra, pois deve permitir que o mesmo seja controlado por voz, auxiliando, assim, no processo de registro de medição física de uma obra.

1.1 PROBLEMÁTICA

Criatividade e o ato de inovar, devem fazer parte de um ambiente empresarial, pois estes dois elementos agregam mudanças na forma como o trabalho é executado e proporcionam um diferencial competitivo, quando a ideia é implementada. Porém, segundo Trías de Bes e outros (2012, p. 15), "Embora 96% dos executivos considerem a criatividade essencial para suas empresas, surpreendentemente, apenas 23% deles tiveram êxito em torná-la parte integrante da empresa". Hoje, empresas ainda utilizam meios de gerenciamento arcaicos, pois são procedimentos padrões e satisfatório no sentindo de não arriscar, mas para uma visão de mercado, uma inovação agregaria muito valor, tanto para os processos internos, quanto para processos externos, ligados aos clientes. (SILVA, 2003, p. 19)

Em relação ao que está sendo proposto neste trabalho, quando se olha para uma obra no contexto da construção civil, pode-se notar que existe uma série de processos e tarefas executadas por várias equipes que além de responsáveis por manter a qualidade do serviço são responsáveis também pela gestão dos custos da obra. Estas tarefas são realizadas diariamente e junto ao canteiro da obra. Estas equipes precisam fazer a coleta de diversas informações

[1] Sienge: Explicação no capítulo 2, seção 2.1.

pertinentes a estes controles, e tudo é registrado em planilhas ou em aplicativos mobile, para posterior integração da informação com o ERP[2] Sienge.

Quando se observa como está sendo feito o controle interno e medições da obra, verifica-se que um colaborador realizando as medições, onde precisa usar diversos artefatos como, por exemplo, fita métrica e precisa memorizar esses dados para posteriormente terminar a avaliação das medidas, assim como demanda tempo adicional para examinar as variáveis e depois anotá-las.

Ao medir o levantamento de uma parede ou tamanho construído de uma calçada, por exemplo, o colaborador responsável precisa medir quanto daquela atividade foi realizada naquele dia, dependendo das tarefas que ele executou, ou está executando, ele pode estar utilizando luvas ou estar com as mãos sujas, o que impossibilita o manuseio do sistema através do dispositivo móvel. Os dados das medições precisam ser anotados em uma planilha temporária para serem passados posteriormente para o sistema memorizá-los, para que depois possa armazená-los.

Tendo isso, pensou-se em um aprimoramento para o módulo de registro de medição física da obra do sistema, que contará com um assistente controlado por voz para auxiliar na inserção das informações. Então, enquanto o funcionário exerce suas funções, ele já poderá realizar o registro de dados significativos do que já foi feito e do que ainda está sendo feito, com isso ele poderia agilizar o processo de medição, assim como, diminuir consideravelmente os erros de escrita ou esquecimento de informações que gerariam retrabalho.

1.2 OBJETIVOS

A seguir, são apresentados os objetivos deste trabalho.

[2] ERP – Planejamento de Recurso Corporativo: Explicação no capítulo 2, seção 2.6.

1.2.1 Objetivo geral

O presente trabalho tem por objetivo o desenvolvimento de um sistema móvel de auxílio para o registro de informações em canteiros de obras via comando de voz.

Espera-se que este sistema aumente a experiência de uso facilitando a navegação entre as tarefas da sua obra e consequentemente agilizando o processo de registro de informações em canteiros de obras.

1.2.2 Objetivos específicos

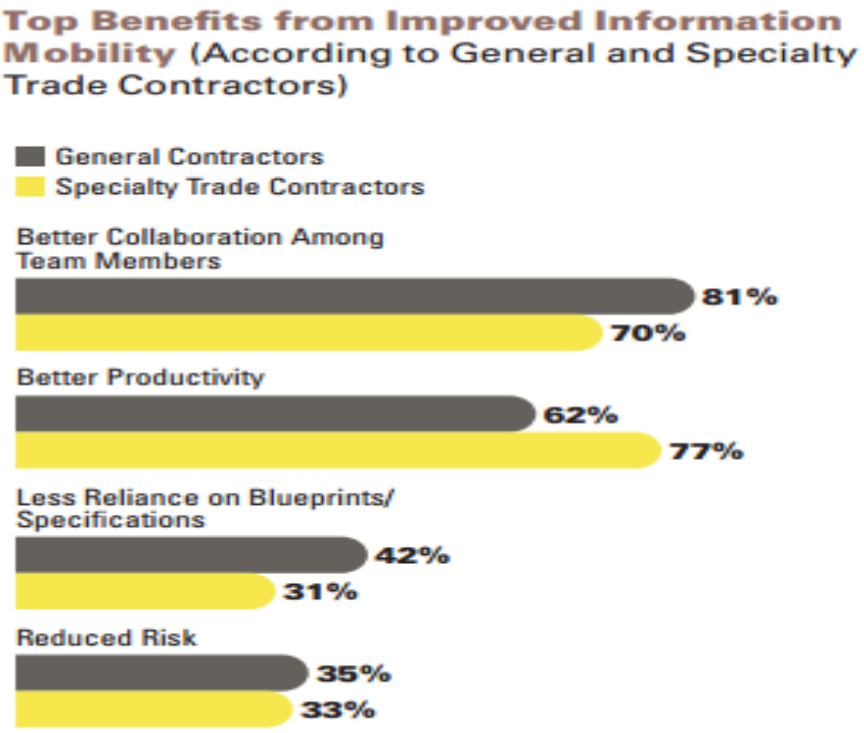
O presente trabalho tem como objetivo específico, os seguintes itens:

- Realizar pesquisas de sistemas e bibliotecas para reconhecimento de voz.
- Analisar os processos relacionados com a rotina do gestor de mão de obra.
- Modelar um protótipo funcional de aplicação móvel.
- Desenvolver este protótipo.
- Avaliar o sistema a partir de um experimento aplicado.

1.3 JUSTIFICATIVA

Bernstein e Laquidara-Carr (2013) descrevem que avanços na troca de informações têm ajudado na melhoria da produtividade de indústria da construção, no entanto, muitas empresas ainda estão experimentando desafios, especialmente com a melhoria do fluxo de informações no canteiro da obra, um fator fundamental para o aumento da produtividade. Levando isso em consideração, podemos afirmar que sistemas com o objetivo de facilitar o fluxo de informações no canteiro da obra devem ocasionar um aumento na produtividade. Demonstrativo do aumento significativo na Figura 1:

Figura 1 – Top Benefits from Improved Information Mobility



Fonte: McGraw Hill Construction, 2013

O gráfico revela que a mobilidade da informação em obras cujo o contrato é um contrato geral houve um ganho de (62%) em produtividade e em contratos específicos cujo o objetivo tem alguma especialidade ou particularidade o aumento de produtividade foi ainda maior (77%), além de outros ganhos como por exemplo redução de risco ou em outras palavras confiabilidade da informação (35%) em ambos os casos quando têm-se esse percentual aplicado ao mercado da indústria da construção no brasil que movimenta bilhões de reais por ano, um número interessante começa a surgir. Segundo a pesquisa anual da indústria da construção, as empresas de construção em 2012 realizaram incorporações, obras e serviços no valor de R$ 336,6 bilhões. (IBGE, 2012).

Com isto, ferramentas com o objetivo de facilitar o fluxo de informações no canteiro da obra tem parte importante no desenvolvimento sustentável de uma construção, parte do controle e medição da obra é feita pelo colaborador junto com o Sienge, porém a manipulação dos dados se torna complexa pois o colaborador precisar manusear o dispositivo durante as medições cujo momento está ocupado realizando as mesmas com artefatos de medição (fitas métricas ou dispositivos infravermelhos), neste caso ele pode optar por memorizar as informações e inseri-las posteriormente, diminuindo a confiabilidade da informação. Outra opção seria disponibilizar um assistente para ajudá-lo na tarefa o que vai

contra o aumento na produtividade pois é um funcionário a menos que poderia estar realizando outra tarefa. A justificativa do desenvolvimento deste trabalho é facilitar o fluxo de informações na obra de forma que este resulte em um aumento na produtividade e na confiabilidade da informação, neste cenário é que entra o comando de voz que pode ser acionado e controlado em meio a outras atividades pertinentes à medição e facilitando para o usuário a manipulação dos dados. Além disso, no momento da medição, o usuário pode inserir a informação via comando de voz no sistema sem necessitar memorizar ou anotar em alguma planilha para posteriormente cadastrá-las, o que aumenta a confiabilidade na informação.

1.4 ESTRUTURA

O presente trabalho está estruturado da seguinte forma:

- Capítulo 1 – Introdução: Este capítulo apresenta a introdução, problema, os objetivos e a justificativa do trabalho.
- Capítulo 2 – Revisão Bibliográfica: Nesse capítulo é referenciado o conteúdo que se faz necessário para a fundamentação teórica que embasa o projeto.
- Capítulo 3 – Metodologia: Apresentação da metodologia de pesquisa, a proposta da solução e as delimitações do projeto.
- Capítulo 4 – Modelagem: Apresentação dos modelos e diagramas do sistema.
- Capítulo 5 – Desenvolvimento: Nesse capítulo, apresenta um conjunto de informações de como foi feito o desenvolvimento do sistema controlado por voz, bem como avaliação do sistema.
- Capítulo 6 – Conclusão e trabalhos futuros: Este capítulo apresenta as conclusões deste trabalho e sugestões para trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

Neste capítulo, são abordados os processos relacionados com o dia a dia de uma obra, assim como os conhecimentos sobre a aplicação, sistema e conceitos.

2.1 SIENGE

Segundo a descrição do produto no site institucional (http://www.sienge.com.br/o-sienge), o Sienge[3] é um sistema de gestão, também chamado de *ERP – Enterprise Resource Planning*, especializado na Indústria da Construção. Pode-se, por meio deste, gerenciar e integrar todas as áreas de uma empresa da indústria da construção, sem ter que abrir mão de um software que atenda com propriedade a produção da sua empresa.

O Sienge é desenvolvido para operar inteiramente na web. Pode ser acessado por computadores, smartphones, notebooks e tablets, de forma rápida e simples. Ainda traz como ponto positivo uma redução de investimentos em infraestrutura, ao contrário de softwares Desktop que normalmente exigem altos investimentos em TI. Pode ser acessado pelos sistemas operacionais Linux, Windows, Android e IOS, e foi desenvolvido para operar nos navegadores mais populares: Internet Explorer, Firefox, Google Chrome e Safari.

2.1.1 Sienge acompanhamento

O sistema Acompanhamento facilita o controle da execução da obra através de registros de medições físicas e de relatórios comparativos entre o planejado e o realizado, de forma a permitir uma resposta ágil caso haja atrasos ou imprevistos nas obras.

[3] Sistema desenvolvido e mantido pela empresa de desenvolvimento de software denominada Softplan. (Disponível em: www.sienge.com.br. 2014)

É possível importar percentuais executados a partir de arquivos do MS-Project[5], caso necessário.

As medições podem ser registradas no sistema também a partir de tablets ou smartphones, ou mesmo através de planilhas impressas com layout que facilita o registro, conferência e autorização manual[6].

2.2 SIENGE MOBILE REGISTRO DE MEDIÇÃO FÍSICA

O aplicativo Registro de medição física permite que usuários do ERP Sienge registrem medições utilizando tablets ou smartphones. Estas medições alimentam o andamento físico da obra no Módulo Acompanhamento de Obras do ERP.

O aplicativo permite[7]:

- Sincronizar com o ERP, trazendo as atividades planejadas para o dispositivo e após, registrar as medições sem necessidade de acesso à internet;
- Visualizar as atividades a serem medidas em formato de árvore, agrupados por obra e unidade construtiva (torre, pavimento, frente, etc);
- Informar as quantidades ou percentuais executados para cada tarefa;
- Sincronizar com o ERP, enviando a medição que será registrada automaticamente no módulo Acompanhamento de Obras;
  Utilização em IOS e Android.

2.2.1 Tecnologias semelhantes

Existem tecnologias capazes de realizar a medição e gerar dados dessa medição automaticamente sem que um usuário precise manualmente medi-la e registrá-la em algum

[5] Software desenvolvido pela Microsoft, para elaboração e acompanhamento de projetos.

[6] Informações obtidas pelos autores, através do conhecimento adquirido pelo uso do sistema.

[7] Comunicação verbal obtida de Luciano Cervo (Softplan, 2015).

sistema, por exemplo, há um aplicativo para smartphones que possuem a plataforma iOS (sistema operacional desenvolvido pela empresa Apple) chamado *Roomscan* (desenvolvido pela empresa Locometric). Este aplicativo permite que os usuários tirem medidas de qualquer tipo de ambiente ao caminhar por seu perímetro, gerando instantaneamente uma planta baixa com margem de erro de no máximo 5 centímetros para mais ou para menos.

Os autores deste trabalho consideram que, fazendo uma analogia com o sistema de registro de medição de mão de obra explicado nos parágrafos anteriores seria uma tecnologia muito interessante para área da indústria da construção e que ajudaria muito no registro de medição física de mão de obra, por exemplo, para informar o tamanho das paredes do ambiente, para o registro da medição sem que o profissional precise medir parede por parede manualmente com algum tipo de artefato.

2.3 ANDROID

Android Inc foi um *startup* criada por Andy Rubin em Palo Alto (Califórnia – USA), onde seu objetivo era o desenvolvimento de uma plataforma de dispositivos móveis baseada em kernel Linux[8]. Em meados de 2005, uma empresa, chamada Google, se interessou pelo o que foi proposto por Andy Rubin e comprou esta *startup*. Ambos os envolvidos tinham a ideia de fazer um sistema operacional não apenas para um único fabricante de hardware, mas um sistema que qualquer um pudesse ter a licença e colocar em seu aparelho, também pensavam que deveria ser flexível e atualizável. (HILL, 2010).

Assim, em 2007, para poder ter mais atenção e continuidade com sucesso ao projeto Android, o Google criou a *Open Handset Alliance* (OHA), um consórcio de empresas de tecnologia envolvidas no mercado de dispositivos móveis (HTC, Sony, Intel, Motorola, Samsung, etc). No fim do ano de 2007, é liberada a primeira versão do Android, juntamente com o SDK para o desenvolvimento de aplicações. (PRADO, 2011, p. 1).

Para desenvolvimento destas aplicações que executam no Android, foi escolhida a linguagem Java, a linguagem de programação mais popular no mundo. (FILHO, 2014, p. 17).

Versões de Android ao longo dos anos[9]:

[8] Estrutura base do sistema operacional.

[9] Fonte: Próprio site do Android: https://www.android.com/intl/pt-BR_br/history/

- Android 1.6: Donut;
- Android 2.0: Eclair;
- Android 2.2: Froyo;
- Android 2.3: Gingerbread;
- Android 3.0: Honeycomb;
- Android 4.0: Ice Cream Sandwich;
- Android 4.1: Jelly Bean;
- Android 4.4: KitKat;
- Android 5.0: Lollipop;
- Android 6.0: Marshmallow;

2.3.1 Plataforma Android

O Android foi a primeira plataforma de desenvolvimento móvel *Open-Source*[10] baseada em Java com sistema operacional Linux, onde todas as suas características foram incorporadas ao Android, bem como sistemas de arquivos, Kernel e os servidores de terminal.

Conforme descrito por Ableson e outros (2012, p. 4), a plataforma Android suporta algumas características, como:

> O Android inclui um Sistema Operacional (OS) baseado em um kernel Linux, uma rica Interface de Usuário (IU), aplicativos de usuário, bibliotecas de código, frameworks de aplicativo, suporte a multimídia e muito mais. E sim, até funcionalidades de telefone estão incluídas [...]

Android têm suas particularidades perante os outros sistemas para dispositivos móveis, a qual o dá uma certa vantagem em relação a seus concorrentes no mercado. Conforme Strickland (2008):

> Um fator importante que posiciona o Android distante da maioria dos sistemas operacionais móveis é que ele está baseado em uma plataforma de código aberto. Isso significa que o Google permite a qualquer um olhar e modificar a maior parte do código fonte do Android. Idealmente, isso significa que se um desenvolvedor sentisse que o Android precisasse de um recurso ou capacidade específica, ele ou ela poderia construí-la e incorporá-la ao sistema operacional. O software evoluiria constantemente.

[10] Open-Source: Se refere a algo que pode ser modificado porque seu design é acessível ao público.

O ponto positivo desta questão é que os usuários finais podem realizar os ajustes ou inovações, proporcionando avanços, pois todos estão ajudando a construir algo melhor.

Para Pereira e outros (2009, p. 3), “por ser open source, pode ser sempre adaptado a fim de incorporar novas tecnologias, [...] já que as comunidades de desenvolvedores estarão em conjunto para construir aplicações móveis inovadoras[...]”.

Android é um sistema encontrado em diversos dispositivos, pois é uma aplicação que possui um pouco de tudo, o qual suporta uma grande variedade de dispositivos, entre smartphones, tablets e entre outros. Por isso que muitas empresas que lançam apenas o dispositivo, adotam o Android como seu sistema operacional, pelo encaixamento que o mesmo possui. (ABLESON et al., 2012)

Figura 2 – Sistema Android em dispositivos móveis

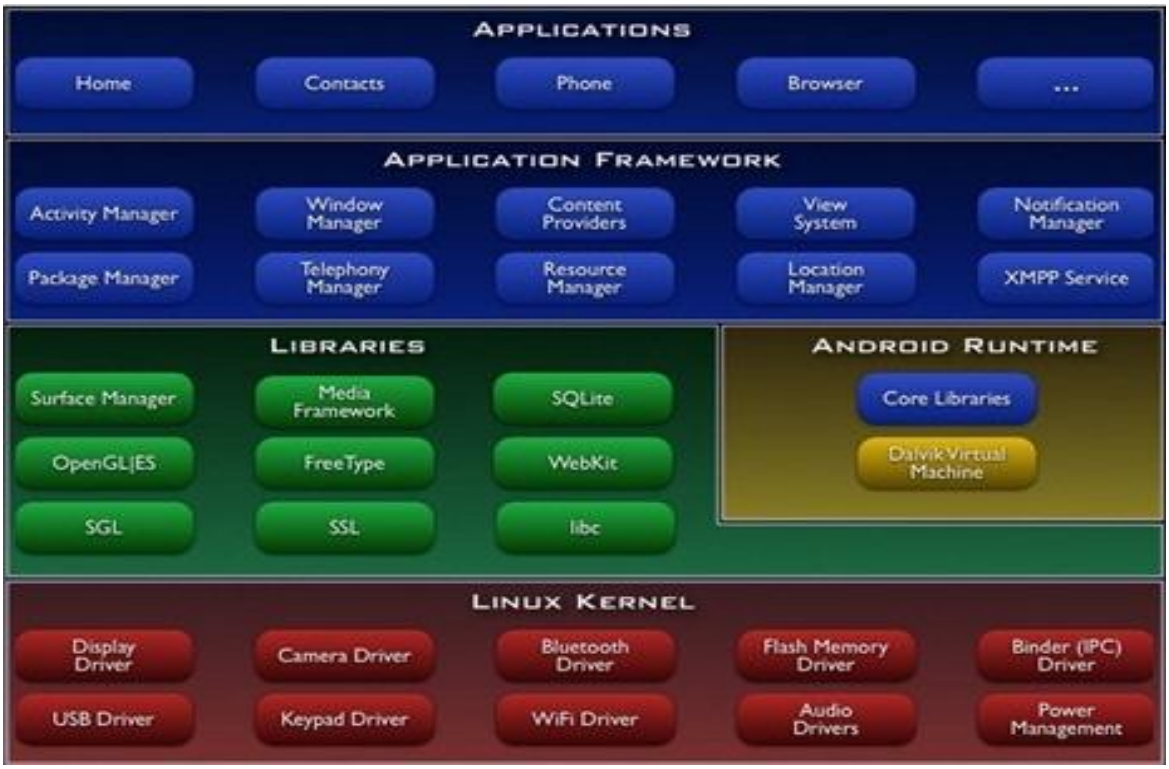
Fonte: http://pbinfor.com.br/blog/o-que-e-um-android/ (2012, p. 1)

O objetivo do Android segue na mesma maneira de outros sistemas operacionais, como Windows da Microsoft, Mac OS da Apple, Ubuntu baseado em Linux, entre outros, cuja função quando em funcionamento é de gerenciar todos os processos dos aplicativos e do hardware de um dispositivo para que funcionem perfeitamente e como esperado.

2.3.2 Arquitetura Android

A arquitetura Android possui basicamente 4 camadas, conforme é mostrado na figura 3. A seguir, breve descrição das mesmas:

Figura 3 – Representação da arquitetura do Android



Fonte: http://androidms.wordpress.com/tag/arquitetura/ (2011, p. 1)

A arquitetura do sistema operacional Android, conforme demonstrado na figura acima tem-se suas camadas divididas, e em cada uma destas camadas encontra-se processos que serão gerenciados dependendo da demanda solicitada.

- Camada de aplicações: Nesta encontram-se todos os aplicativos fundamentais, que para Pereira e outros (2009, p.6) são "cliente de e-mail, mapas, navegadores, calendários, programas de SMS, gerenciador de contatos, agendas, entre outros que serão desenvolvidos pela comunidade".
- Camada de framework: a camada abaixo da camada de aplicações. Para Pereira e outros (2009, p. 6):

> Na camada framework, encontramos todas as APIs[20] e os recursos utilizados pelos aplicativos, como classes visuais, que incluem listas, grades, caixas de textos, botões e até um navegador web embutido, View system (componentes utilizados na construção de aplicativos), provedor de conteúdo, que possibilita que uma aplicação possa acessar informações de outra aplicação, ou até mesmo compartilharem as suas informações, possibilitando a troca de informações entre aplicativos e gerenciadores de recursos (permite definir e carregar recursos em tempo real), gerenciador de localização (GPS e Cell ID), gerenciador de notificação (fornece informações sobre eventos que ocorrem no dispositivo), de pacotes e de atividades, que controla todo o ciclo de vida da aplicação e o acesso e navegação entre as aplicações.

- Na camada de bibliotecas, encontram-se pacotes de códigos já desenvolvidos e um agrupamento de funcionalidades para utilizar no desenvolvimento do aplicativo. (PEREIRA et al, 2009, p. 7).
- A camada Android RunTime, onde na imagem é demonstrada como menor que as outras, é uma estrutura que permite a execução de várias máquinas simultaneamente, proporcionando maior desempenho para a aplicação. (PEREIRA et al, 2009, p. 8).
- Linux Kernel, última camada apresentada na arquitetura do Android, para Pereira e outros (2009, p.9) apoia "os serviços centrais do sistema, tais como segurança, gestão de memória, gestão de processos, pilha de protocolos de rede e modelo de drives".

2.4 INTERFACES MULTIMODAIS

Sistemas de interfaces multimodais processam vários métodos de entrada, gerando, assim, o comando para atender à solicitação do usuário. Então, no processo de criação de um produto interativo, deve-se dar atenção ao planejamento de como projetar a interface física e quais tecnologias e estilos de interação utilizar, por exemplo, multitoque (multitouch), interfaces baseadas na fala (com voz), interface gráfica, head-up display, realidade aumentada, interfaces gestuais e entre outros, pois estes, influenciarão nos métodos de entrada do produto (PREECE, 2011, p. 36).

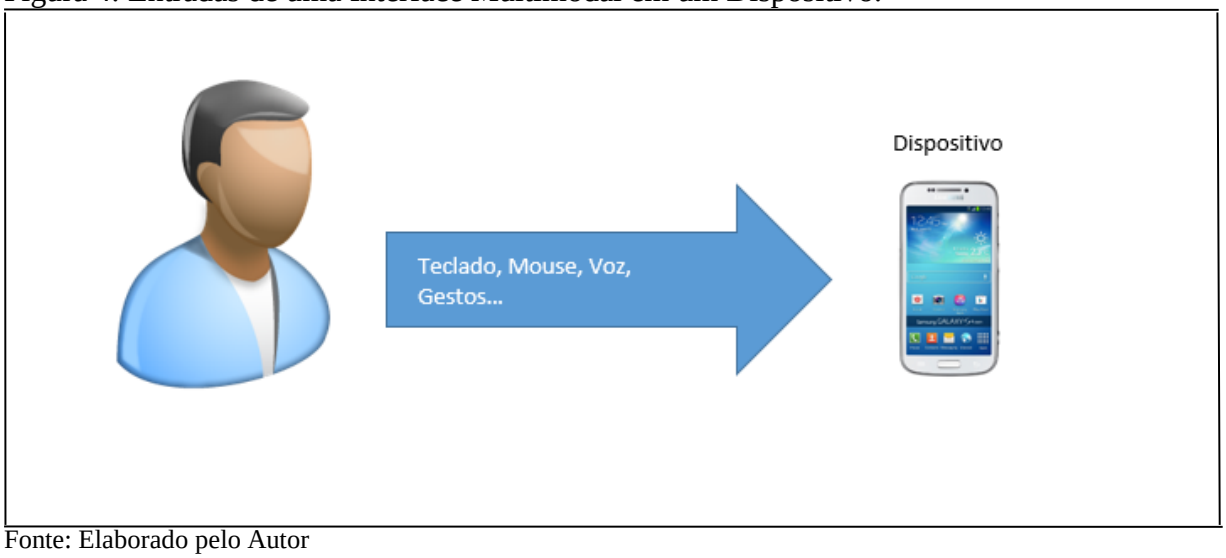
O objetivo das interfaces multimodais é o de apoiar e acomodar os usuários em sua capacidade perceptiva e comunicativa, também, o de integrar habilidades computacionais no

[20] API: Explicação no capítulo 2, seção 2.7.

mundo real, ou seja, oferecer maneiras mais naturais de interação entre humano-computador (LALANNE et al, 2009, p. 5).

Em interfaces multimodais, tem-se diversas entradas em um dispositivo, conforme figura 4, desde que este tenha os devidos hardwares, diferentemente de interfaces comuns, onde a requisição solicitada por um usuário é realizada por um botão clicado ou uma tecla apertada.

Figura 4: Entradas de uma Interface Multimodal em um Dispositivo.



Fonte: Elaborado pelo Autor

Conforme demonstrado na figura acima, há uma relação entre o usuário e o dispositivo, que é o aguardo do comando, podendo ser por teclado, mouse, voz, gestos e outros.

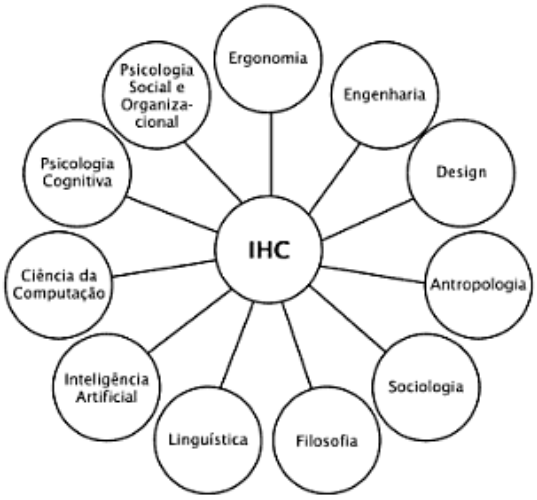
2.4.1 Interação humano-computador

Para Andrade (2007, p. 36) "O termo Interação Homem-Computador (IHC) nasceu em meados dos anos 80, abrangendo muito mais do que o projeto de interfaces, relacionando tudo que estiver envolvido na interação entre usuários e computadores, seja aspectos físicos, psicológicos, práticas de trabalho, relações sociais, saúde, etc."

Para proporcionar uma boa qualidade de interação, estabelecendo os devidos critérios de usabilidade para o mesmo, deve-se ter conhecimento dos aspectos descritos a cima e das

áreas que a IHC engloba, pois assim, tem-se um conhecimento do todo a ser desenvolvido e não apenas a escrita do código. (MILETTO et al, 2014, p. 51)

Figura 5: Disciplinas relacionadas à IHC



Fonte: Andrade (2007, p. 36)

IHC é uma área que vincula algo humano com algo computacional, conforme descrito por Miletto e outros (2014, p. 51):

> Interação Humano-Computador (IHC) é o nome da área que pesquisa o design, a implementação e a avaliação de sistemas interativos no contexto das atividades dos usuários. IHC é eminentemente multidisciplinar e tem intersecções com áreas como ergonomia, design gráfico, informática, linguística semiótica e psicologia cognitiva.

Esta interação entre humano e computador, bem como os processos de desenvolvimento, que proporcionam em uma ligação positiva entre estes dois elementos, o ser humano e a máquina, podem ser visualizados na representação da figura 6:

Figura 6: Objetos de estudo em IHC

Fonte: Barbosa e outros (2010, p. 10)

Conforme demonstrado na figura 6, a interação que ocorre entre pessoas e computadores, deve ser simples e intuitiva, no sentido de maximizar a qualidade de uso desta interação.

2.4.2 Características interface multimodais

Realizando uma comparação com outros tipos de interação entre humano e computador, conforme descrito por Lalanne e outros (2009, p. 5):

> Interação multimodal busca oferecer aos usuários uma interação mais natural e transparente, utilizando a voz, gestos, direção do olhar, etc. Portanto, interfaces multimodais são esperadas para oferecer maneiras mais fáceis, mais expressivamente poderosos e mais intuitivas para usar computadores. Sistemas multimodais têm o potencial de melhorar a interação humano-computador de várias maneiras: reforçada robustez, devido à combinação de diferentes fontes de informações parciais, personalização flexível com base no usuário e contexto e nova funcionalidade envolvendo multiusuário e interação móvel.

Se comparar interfaces multimodais com interface gráfica do utilizador, conhecida como GUI, é possível apresentar certas diferenças, conforme demonstrado no quadro 1:

Quadro 1: Diferença entre GUIs e Interfaces Multimodais

GUI	Interfaces Multimodais
Fluxo de entrada única	Vários fluxos de entrada
Atômico, determinista	Contínua, probabilística
Processamento sequencial	Processamento paralelo
Arquiteturas centralizadas	Distribuído e arquiteturas sensíveis ao tempo

Fonte: Lalanne e outros (2009, p. 6)

Nesta comparação, verifica-se como os fluxos de entrada acontecem nestas modalidades, a forma como a interface é estruturada e processada, bem como a arquitetura.

2.5 RECONHECIMENTO DE VOZ

Torna-se algo simples a inserção de dúvidas e de dados em dispositivos, isto, pela vasta gama de tecnologias que possuímos para realizar estas inclusões ou solicitações, podendo ser por meio de vários métodos, como mouse, scanner, caneta óptica, reconhecimento de caligrafia e de voz. Em relação ao reconhecimento da fala é o método mais apropriado para entrada de dados, edição de textos e computação por conversação, pois interpretará o contexto inserido da fala humana e o discurso realizado entre duas ou mais pessoas é o meio mais cômodo e natural de comunicação humana (SANTOS, 2008, p. 30).

O processamento de linguagem natural, também conhecido como PLN, é o processamento da comunicação que houver entre um usuário e computador, independentemente do idioma em que estiver sendo falado. Ou seja, um computador, a partir de uma solicitação realizada por voz, deve saber como analisar e interpretar esta entrada, podendo incluir conhecimento linguístico das palavras, conhecimento de domínio, conhecimento de senso comum e até mesmo conhecimento dos usuários e seus objetivos, tudo embarcado na análise e interpretação que o sistema deverá realizar. Assim, que a interpretação for finalizada, o sistema então, poderá realizar a ação desejada. Dois tipos de PLN para realizar o descrito acima, primeiro o entendimento da linguagem natural, que investiga métodos que permitem a um computador compreender dadas instruções, via teclado ou por voz, de modo que computadores sejam capazes de entender as pessoas. E por final, a geração de linguagem natural, de nada adianta entender sem dar uma resposta para o usuário que realizou uma solicitação qualquer, assim, o computador produzirá seu entendimento da

solicitação pela tela ou por síntese de voz, para que as pessoas possam entender os computadores mais facilmente (TURBAN et al, 2008, p. 516).

Para Turban e outros (2008, p. 517):

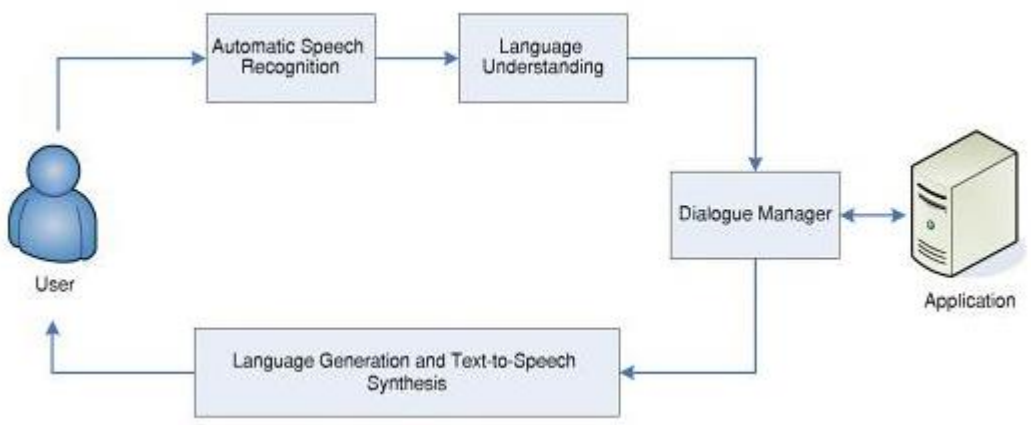
> O reconhecimento da fala é um processo que permite comunicar-se com um computador falando com ele. O termo reconhecimento da fala às vezes só é aplicado à primeira parte do processo de comunicação, no qual o computador reconhece palavras em frases curtas que foram faladas sem necessariamente interpretar os significados. A outra parte do processo, no qual o significado da fala é determinado, é chamado entendimento da fala. É possível entender o significado de uma frase falada sem na verdade reconhecer cada palavra. Quando um sistema de reconhecimento de fala é combinado com um sistema de processamento de linguagem natural, o resultado é um sistema geral que não apenas reconhece entrada de voz, mas também a entende.

Assim que realizado o comando de leitura da fala disponibilizada por um usuário, o sistema realiza certas tratativas, conforme descrito por Ross (2008, p. 70):

> Para a captura do som, o usuário posiciona-se diante de um microfone e pronuncia uma frase previamente selecionada, ou uma frase qualquer. Este processo é repetido várias vezes até que seja possível criar um modelo. Todos os sistemas que analisam a voz estão amplamente baseados na tecnologia de processamento de fala. A forma da onda das frases é medida usando-se análises de Fourier para encontrar o espectro de frequências que mostram as características da voz.

Assim, com a captura da fala, o sistema processa, conforme descrito acima, e converte em um arquivo de processador de texto, para que possa realizar a devida lógica de como se comportar com o que foi falado. A seguir figura 7 retratando o processo:

Figura 7 – Processo Reconhecimento de Voz



Fonte: Schmitt e outros (2013, p. 6)

Sistemas de reconhecimento de voz, aguardam até que o usuário fale com o sistema, assim, começa o reconhecimento, para identificar qual comando foi solicitado. No momento em que houve a identificação, o sistema executa a solicitação, retornando uma mensagem ao usuário e assim segue até finalizar a solicitação por completo, conforme demonstrado na figura 7.

2.5.1 Vantagens e desvantagens do reconhecimento e entendimento da fala

O reconhecimento de fala proporciona vantagens, quando usado, como: Facilidade de acesso, pois ao invés de ficar digitando, simplesmente há uma conversa com o dispositivo. Velocidade de comunicação entre ser humano e máquina, tanto pelo fato de não digitar, quanto pela precisão das informações captadas, também se tem a liberdade manual, deixando de utilizar as duas mãos no dispositivo para utilizar somente a voz. Geração de relatórios pela fala, quando um banco de dados possuir capacidade de reconhecimento de voz e acesso remoto. Por final, a fala é algo único da pessoa, assim como uma digital de um dedo, olho de um ser humano, proporcionando uma segurança, quando implementado algo para restringir por estes mecanismos (TURBAN et al, 2008).

Porém, não possui somente vantagens o reconhecimento e entendimento da fala, há certas desvantagens na sua forma de uso também, como: Ambientes com ruídos podem dissipar a informação falada pelo usuário, fazendo com que o sistema não consiga captar toda esta mensagem. Quando um usuário começa a enviar a informação ao aplicativo, caso esta mensagem seja muito extensa, o sistema acaba não reconhecendo por totalidade a mensagem e por fim, há uma limitação no reconhecimento e entendimento da fala, que é a não manipulação de ícones e imagens presentes na interface do aplicativo, fazendo com que o usuário utilize algum mecanismo para clicar nestes objetos (TURBAN et al, 2008).

2.5.2 Técnicas de reconhecimento de voz

Uma das técnicas de reconhecimento de voz, são as RNAs, também conhecidas como redes neurais artificiais, que consistem basicamente na cópia de um processo de aprendizagem de um cérebro de um ser vivo e, assim, processará as entradas em um sistema, envolvendo dados não lineares (PIO, 2009). “As redes neurais foram desenvolvidas pelo neurofisiologista McCulloch e pelo matemático Walter Pitts da Universidade de Illinois, fazendo uma analogia entre células nervosas e o processo em um artigo intitulado ‘A Logical Calculus of Ideas Immanet in Nervous Activity’ em 1943” (PIO, 2009, p. 3).

2.5.2.1 Estrutura de uma RNA

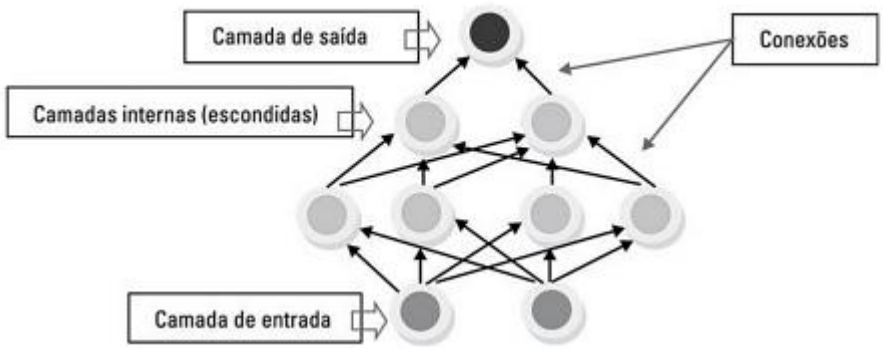
Segundo Pontes (2011, p.59) “Uma RNA é um conjunto de processadores dispostos em forma de rede. Dessa forma, cada um dos nós da rede é capaz de receber informações de outros nós, efetuar algum tipo de processamento e enviar o resultado obtido para outros nós da rede”.

Complementando o descritivo a cima , para Lima et al (2014, 52):

> Uma RNA é um modelo matemático simplificado de um neurônio biológico, que consiste na conexão de várias unidades básicas denominadas elementos de processamento. Estes elementos são organizados em camadas, onde cada elemento tem conexões ponderadas com outros elementos. Essas conexões podem tomar diferentes configurações, dependendo da aplicação desejada. Uma RNA é, geralmente, constituída de uma camada de entrada, que recebe os estímulos do modelo, e de uma camada de saída, que produz a resposta correspondente. Algumas redes podem ter uma ou mais camadas internas (também chamadas ocultas ou escondidas). A capacidade computacional de uma rede neural está nas conexões entre os elementos processadores. Nos pesos que ponderam cada conexão são armazenadas as informações que a rede “aprendeu”.

Conforme figura 8, segue representação da estrutura de uma RNA:

Figura 8: Representação de uma RNA típica.



Fonte: Lima et al (2014, 52)

A estrutura representada a cima, depende sempre de uma entrada. Assim que for realizada uma entrada na estrutura, a mesma passa para as camadas escondidas para realização dos cálculos e das operações para se chegar no valor de saída. Este, apresenta uma resposta final do conjunto das informações inseridas no conhecimento da rede neural, ou seja, na entrada.

Para estruturar uma RNA, a mesma deve ser desenvolvida a partir do índice que se quer chegar, ou seja, a resposta ao problema da situação ocorrida. Assim, desenham-se camadas internas, para que possa realizar os devidos testes para solucionar ou para simplesmente apresentar um dado ou um conjunto de informações na camada de saída (PONTES, 2011).

2.6 ERP

Segundo Junior (2008, p. 86) “Começaram a ser utilizados mundialmente no início da década de 1990. No Brasil, as primeiras implementações ocorreram por volta de 1997 e 1998. Devido ao seu alto valor, eram viáveis apenas às grandes corporações e multifuncionais”.

Um sistema deste nível, que pode proporcionar grandes ganhos a empresa, no sentido de poder organizar a maioria dos processos internos e externos, bem como integrar todas as áreas em um único sistema e realizar todo e qualquer inserção de dados da empresa diariamente, tanto para registro, quanto para futuro geração de relatórios, tem um custo alto,

por isso começou a crescer mais tarde e muito em momentos que as empresas precisavam inovar (BALTZAN, 2012).

Para Santos (2008, p. 69):

> Os sistemas ERP devem concentrar-se no apoio a processos empresariais envolvidos nas operações de uma empresa. Todas as áreas de negócio devem estar envolvidas e com foco na satisfação dos clientes, dos colaboradores, na sintonia com as novas tecnologias, maximização do uso dos recursos e no alinhamento destes e do estilo único dos gerentes. O sistema ERP abrange vários programas que apoiam as atividades empresariais nos processos de negócios de marketing, produção, logística, recursos humanos, contabilidade e finanças de uma empresa. Hoje o ERP passou a ser um ingrediente para reduzir riscos e falhas, alinhar processos e necessidades e otimizar produtividade e os resultados. As empresas, como All Tasks, atribuem grande valor ao ERP por dois motivos, acompanhar todas as etapas do projeto em tempo real e informações vitais em relatórios permitem avaliar o desempenho da empresa.

Ainda há empresas que ficam nas mãos de desenvolvedores terceiros, montando várias e várias aplicações diferentes, sem nenhuma integração, enquanto um ERP solucionaria vários problemas internos, bem como uma padronização única.

2.6.1 Principais características e vantagens do ERP

Um ERP possui características notáveis, que o fazem ser escolhido pelas empresas para realizar a parte de apoio a todos os processos que envolvem esta organização, as quais são: Possui banco de dados único, isto, para integrar as diversas áreas funcionais, compartilhado por toda a empresa. São pacotes comerciais, ou seja, possuem uma estrutura interna padrão adaptável a diversos segmentos de negócios em que a empresa estiver alocada. Apresenta estrutura modular, que é a separação do ERP por módulos, no sentido de que cada área funcional é um módulo dentro deste ERP, compartilhando uma mesma base de dados. É desenvolvido com base nas Best Practices, que é quando um ERP é projetado, realiza-se então, um estudo de mercado para identificar as melhores práticas aplicadas a cada segmento, ou seja, um ERP já passou por praticamente toda otimização possível daquele processo em questão e isto, trará vantagens competitivas para a organização que está a implementar o sistema (JUNIOR, 2008, p. 89).

Este produto pode trazer à empresa inúmeras vantagens, quando este, é utilizado de maneira correta, ou seja, absorvendo todas as funcionalidades para maximizar as vantagens competitivas da empresa. Vantagens: Elimina redundância e redigitação de dados, isto, pelo fato de possuir um banco de dados único, todas as informações estarão inseridas nele e compartilhadas para consulta, acabando com dados duplicados e o retrabalho na sua inserção. Possibilita maior integridade das informações, pois, quando realiza-se a alteração de alguma informação no ERP, esta alteração é atualizada para todos os módulos que fazem consulta em cima deste dado, deixando sempre sincronizado e atualizado. Aumenta a segurança sobre os processos de negócios, pois possuem controles de permissões de acesso. Permite rastreabilidade de transações já que deve possuir credenciais de acesso, toda e qualquer alteração efetuada é armazenada em um arquivo, chamado de log, onde este, é passível de auditoria. Usando um ERP, terá uma padronização de sistemas, quando este, for bem aplicado à sua empresa, ou seja, suprir todos os outros sistemas existentes pelos módulos que o ERP dispõe (JUNIOR, 2008, p. 90).

2.7 API

API, interface de programação de aplicativos, é uma coleção de componentes de software, ou seja, um conjunto de classes já desenvolvidas tanto pelos criadores da linguagem de programação que estiver a utilizar quanto pelos amantes de programação de software e, este conjunto de classes, podem ser utilizados por desenvolvedores para escreverem seus próprios aplicativos. Estes componentes estão agrupados e organizados em pacotes, contendo conjunto de instruções e padrões de programação. (DEITEL et al, 2010, p. 1071)

Existe uma especificação da API em formato de documentação, que pode ser encontrada no próprio site[23]. “A documentação apresenta uma visão geral de todas as classes e interfaces, resume seus membros (isto é, os campos, construtores e métodos das classes e os campos e métodos das interfaces) e fornece descrições detalhadas sobre cada membro” (DEITEL et al, 2010, p. 1071).

[23] Site: http://docs.oracle.com/javase/7/docs/api/

2.7.1 Pesquisando a API

Muitos programadores mundo a fora, quando estão realizando o desenvolvimento de um novo aplicativo, obtém auxílios com a documentação da API.

Para Deitel e outros (2010, p. 1071):

> Os programadores pesquisam a API para encontrar o seguinte: O pacote que contém uma classe ou interface particular. Os relacionamentos entre uma classe ou interface particular e outras classes e interfaces. Classes ou constantes de interface, normalmente declaradas como campos public static final. Construtores para determinar como um objeto da classe pode ser inicializado. Os métodos de uma classe para determinar se eles são static ou não static, os números e tipos dos argumentos que você precisa passar, os tipos de retorno e quaisquer exceções que poderiam ser lançadas a partir do método. E, também, os programadores costumam recorrer à documentação para descobrir classes e interfaces que eles ainda não usaram [...].

Estes pacotes encontrados na API, já foram testados e usados por outros desenvolvedores e compartilhados para uso de outros programadores, otimizando seu tempo de desenvolvimento.

2.7.2 Importância da API

Uma API é tão importante tanto para os desenvolvedores de uma nova aplicação, quanto para àqueles que já possuem sua aplicação no mercado, pois a API fornecerá métodos simples de uma aplicação já desenvolvida, por exemplo, Facebook, YouTube e entre outros, para que possa ter uma comunicação entre esta aplicação já desenvolvida com àquela a ser desenvolvida.

Complementado, “Uma API define as regras de comunicação e interação com outros programas de dentro de um programa específico. [...] invocarão funcionalidades especiais em coisas como a execução de um áudio, a execução de um vídeo e a interação com outras aplicações” (TITTEL et al, 2014, p. 320).

Estas funcionalidades ou métodos que as aplicações já existentes liberam para novos desenvolvedores aplicarem em seu novo projeto são, por exemplo, a disponibilidade de poder

colocar o botão curtir e compartilhar do Facebook em sua própria página, já se comunicando e interagindo com o Facebook. Outro exemplo, seria a opção de colocar a estrutura e design dos vídeos do YouTube em sua página, já se comunicando e interagindo também com o site do YouTube.

2.8 TECNOLOGIAS UTILIZADAS PARA IMPLEMENTAÇÃO

Nesta seção será apresentado uma revisão tecnológica das tecnologias utilizadas na implantação do protótipo, sendo elas a linguagem de programação, bem como as ferramentas de desenvolvimento e modelagem.

2.8.1 Java

Segundo Sampaio (2007, p.1) "A plataforma Java se estabeleceu como um padrão para desenvolvimento de aplicações corporativas, enquanto outras plataformas perderam seu público, sendo gradativamente substituídas". Isto hoje, mas no seu início, 1991, quando os microprocessadores tiveram um impacto em alguns dispositivos que a sociedade utilizava, a Sun Microsystems financiou uma pesquisa corporativa de James Gosling, onde esta, baseava-se pela linguagem de programação C++, resultando na linguagem Oak. Mais tarde, teve seu nome trocado por Java, quando a equipe da Sun visitou uma cidade de origem de um tipo de café importado e, esta, tinha este mesmo nome, Java (DEITEL et al, 2010, p. 6).

Em 1993, quando a Web explodiu em popularidade, o Java passou por um redesenho de seu projeto, onde pôde contar com a adição de conteúdo dinâmico, como interatividade e animações nas páginas da Web (DEITEL et al, 2010, p. 6).

Assim, para Deitel e outros (2010, p. 6):

> A Sun anunciou o Java formalmente em uma conferência do setor em maio de 1995. O Java chamou a atenção da comunidade de negócios por causa do enorme interesse na Web. O Java é agora utilizado para desenvolver aplicativos corporativos de grande porte, aprimorar a funcionalidade de servidores da Web (os computadores

> que fornecem o conteúdo que vemos em nossos navegadores da Web), fornecer aplicativos para dispositivos voltados para o consumo popular (como telefones celulares, pagers e PDAs) e para muitos outros propósitos.

Como é uma linguagem que pode ser executada em qualquer plataforma, Java é hoje a linguagem utilizada na maioria das universidades, fazendo com que seja a primeira linguagem de programação que um aluno aprende e assim, virando uma linguagem universal (COSTA, 2008).

2.8.1.1 A linguagem Java

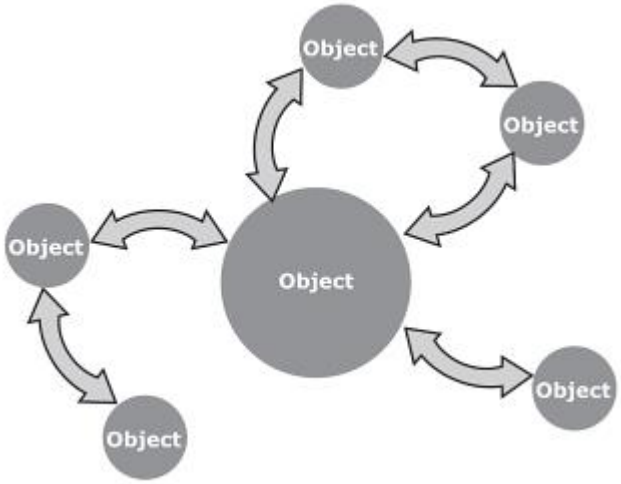
Segundo Somera (2010, p.16) "A linguagem Java utiliza a metodologia orientada a objetos, dando-nos a possibilidade de criar programas flexíveis e modulares, além de reutilizar códigos já criados". Ou seja, em Java são criadas estruturas de código, onde estes, podem ser reutilizados em outras estruturas deste mesmo código da aplicação.

Conforme descrito por Deitel e outros (2010, p. 16):

> No Java, a unidade de programação é a classe, a partir da qual os objetos por fim são instanciados (criados). Classes Java contêm métodos (que implementam operações) bem como campos (que implementam atributos). Programadores em Java concentram-se na criação de classes. Cada classe contém campos e o conjunto de métodos que manipulam os campos e fornecem serviços aos clientes (isto é, outras classes que utiliza a classe). O programador utiliza classes existentes como blocos de construção para construir novas classes.

Exemplificando o que foi descrito a cima, sobre programação orientada a objetos é como se "as classes estão para os objetos assim como as plantas arquitetônicas estão para as casas. Assim como podemos construir muitas casas a partir de uma planta, podemos instanciar (criar) muitos objetos a partir de uma classe" (DEITEL et al, 2010, p. 16). Isto, porque as classes podem ter relacionamentos com as outras classes existentes no código da aplicação por meio de associações, conforme ilustrado na figura 9.

Figura 9: Linguagem orientada a objetos



Fonte: Serson (2007, p. 2)

Conforme representação na figura 9, é apresentada as relações que os objetos de diferentes classes, mas de um mesmo software, possuem.

2.8.1.1.1 Classe

Quando é realizado a definição das classes que irão compor o software e os inter-relacionamentos que devem ocorrer para o devido funcionamento deste software, é descrito quais propriedades ou atributos e comportamentos que os objetos terão. Estes comportamentos são descritos pelos métodos, que por sua vez, definem as funcionalidades da classe (SOMERA, 2006, p. 12).

Um método pode ter três tipos de visibilidade, definindo assim os inter-relacionamentos que devem ter, para que assim, os objetos possam realizar a devida comunicação: Público, o qual deixa visível as informações de uma classe para outros objetos, privado, o qual é o contrário de público, cortando a visibilidade que antes continha, ou seja, as informações de uma classe não estão visíveis para outros objetos. Por fim, há o protegido, o qual apenas deixa ser visível, quando foi criado uma herança entre os objetos das classes, se não, restringe a visibilidade (SOMERA, 2006).

2.8.1.1.2 Objetos

Objetos são instâncias de classes, ou seja, quando é realizado a criação de um objeto, é reservado uma área de memória para realizar o armazenamento dos atributos definidos na classe que foi criada.

Para Somera (2006, p. 14):

> Quando se cria um objeto, esse objeto adquire um espaço em memória para armazenar seu estado (os valores de seu conjunto de atributos, definidos pela classe) e um conjunto de operações que podem ser aplicadas ao objeto (o conjunto de métodos definidos pela classe). Um aplicativo criado com orientação a objetos é composto por um conjunto de objetos que interagem por "trocas de mensagens". Na prática, essa troca de mensagens se traduz na aplicação de métodos a objetos.

Complementando, quando criado o objeto, é a classe quem define o comportamento que este objeto tem no código do sistema a ser desenvolvido.

2.8.1.1.3 Herança

Em orientação a objetos, a chave principal que tornará este tipo de programação único, são as heranças determinadas para compor o desenvolvimento da aplicação.

Para Somera (2006, p. 15):

> O mecanismo de herança permite que características comuns a diversas classes sejam fatoradas em uma classe base, ou superclasse. A partir de uma classe base, outras classes podem ser especificadas. Cada classe derivada, ou subclasse, apresenta as características (estruturas e métodos) da classe base, acrescentando-lhes o que for definido de particular.

Os relacionamentos que a herança dispõe são: Extensão, especificação e combinação, onde cada uma compartilha de alguma forma similaridades entre as classes do código (MATTOS, 2007).

2.8.1.2 Tecnologia Java

O Java é a base para praticamente todos os tipos de aplicações em rede e é o padrão global para o desenvolvimento e distribuição de aplicações móveis e incorporadas, jogos, conteúdo baseado na Web e softwares corporativos[24].

Assim, a tecnologia Java está distribuída em certas áreas, ou seja, onde se encaixa cada tipo de tecnologia e seu desenvolvimento. Assim sendo, Java está organizando com as seguintes tecnolgias: J2SE, J2EE, J2ME, Java Card, Java Web Services, XML e entre outras, as quais vão desde uma plataforma para desenvolvimento do código-fonte pelo programador, até bibliotecas, APIs e tecnologias já implementadas e disponíveis para uso (SOMERA, 2006).

2.8.2 Ferramentas de desenvolvimento e modelagem

Nesta seção são apresentadas as ferramentas utilizadas para realização de todo o desenvolvimento e modelagem do protótipo.

2.8.2.1 Eclipse

Segundo Deitel e outros (2012, p. 70) "O Eclipse permite gerenciar, editar, compilar, executar e depurar aplicativos. O plug-in ADT para Eclipse fornece as ferramentas adicionais necessárias para desenvolver aplicativos Android".

É muito utilizada não só para desenvolvimento de aplicativos Android, mas também de códigos em linguagem Java para outras plataformas, bem como outras linguagens.

Para Deitel e outros (2012, p. 7):

[24] Informações obtidas através do site da Oracle. Disponível em: <https://www.java.com/pt_BR/about/> Acessado 20/05/2015

> Eclipse é uma IDE de código aberto para a construção de programas de computador. O projeto Eclipse foi iniciado na IBM, que desenvolveu a primeira versão do produto e doou-o como software livre para a comunidade. O gasto inicial da IBM no produto foi mais de 40 milhões de dólares. Hoje o eclipse é a IDE Java mais utilizada no mundo. Possui como características marcantes o uso da SWT e não do Swing como biblioteca gráfica, a forte orientação ao desenvolvimento baseado em plug-ins e o amplo suporte ao desenvolvedor com centenas de plug-ins que procura atender às diferentes necessidades de diferentes programadores.

A ferramenta Eclipse aceita outros tipos de linguagem de programação para desenvolvimento de aplicativos, sendo eles: C/C++, PHP e outros. Assim, "cada conjunto de ferramentas do Eclipse que você instala é representado por uma perspectiva de desenvolvimento separada. Modificando-se as perspectivas, o IDE é configurado para usar as ferramentas da linguagem correspondente" (SERSON, 2007, p. 70). Ou seja, assim que for definido o tipo de linguagem a ser utilizada para o desenvolvimento, o Eclipse se ajustará às ferramentas, lógica e semântica da linguagem a ser empregada no código fonte da aplicação.

Concluindo, para Deitel e outros (2012, p. 7):

> IDE, do inglês Integrated Development Enviroment ou Ambiente Integrado de Desenvolvimento, é um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de software com o objetivo de agilizar este processo. Geralmente os IDEs facilitam a técnica de RAD (de Rapid Application Development, ou "Desenvolvimento Rápido de Aplicativos"), que visa a maior produtividade dos desenvolvedores. O Eclipse tronou-se famoso não apenas por ser uma IDE de primeira qualidade, desenvolvido com a extensibilidade em mente, mas também porque ele tinha uma interface gráfica levíssima e que era igual a do sistema operacional onde ele estivesse sendo executado.

O Eclipse possui toda uma interface de apoio para o código a ser desenvolvido, desde apoio para as classes e objetos a serem criados, até toda parte de execução em tempo real e depuração para encontrar erros no código.

2.8.2.1 Enterpise Architect

É uma ferramenta desenvolvida pela Sparx Systems, a qual é utilizada para modelar todo o software a ser desenvolvido, antes da etapa de desenvolvimento, ou seja, é a fase de desenho e estruturação deste software. "As ferramentas de modelagem de análise fornecem a

capacidade de desenvolver modelos baseados em cenários, modelos baseados em classes e modelos comportamentais usando a notação UML" (PRESSMAN, 2011, p. 192).

Para Pressman (2011, p.192):

> As ferramentas nesta categoria suportam todo o conjunto de diagramas UML necessárias para construir um modelo de análise (as ferramentas também dão suporte à modelagem de projeto). Além da diagramação, elas [1] realizam verificação de consistência e correção para todos os diagramas UML, [2] fornecem links para projeto e geração de código, [3] constroem um banco de dados que permite o gerenciamento e a avaliação de grandes modelos UML necessários para sistemas complexos.

Enterprise Architect proporciona aos seus usuários uma série de funcionalidades, sendo elas: possibilidade de se trabalhar com BPMN, sistema totalmente integrado com todas as visões que forem inseridas do software a ser desenvolvido, gerenciamento de requisitos, projetos e testes, manutenção, simulação e depuração, geração de documentos, engenharia do código e entre outros.[25]

2.8.2.1 Android SDK

A primeira versão do Android SDK foi liberada nos anos de 2008, mas a primeira ferramenta de especificação de qualidade dos códigos desenvolvidos para Android foi disponibilizada somente no ano de 2011, chamada de Android Lint, a qual é uma ferramenta que lê e verifica os arquivos de origem do projeto Android para encontrar possíveis erros e otimização de melhorias, proporcionando, assim, segurança, desempenho, usabilidade, acessibilidade e internacionalização. (PAPAPETROU et al, 2015, p.158)

Para Cinar (2012, p.20) "O kit de desenvolvimento do software Android, Android SDK, é o componente central do conjunto de ferramentas de desenvolvimento, fornecendo bibliotecas de API e ferramentas para desenvolvedores que são necessários para a construção, teste e depuração de aplicativos para Android". Ou seja, é uma ferramenta que proporciona um ambiente para desenvolvimento de aplicações para a plataforma móvel Android, do Google.

---

[25] Informações obtidas através do site da Sparx Systems. Disponível em: <http://www.sparxsystems.com.au/products/ea/index.html> Acessado 24/05/2015

2.8.2.2 API utilizadas na aplicação

"Text-To-Speech (TTS) é a API responsável por converter texto em voz e fazer o dispositivo se comunicar com o usuário através de áudio falar, e Speech-To-Text (STT) é o responsável por converter voz em texto e fazer o Android ouvir" (LECHETA, 2013, p. 784). Estas duas APIs foram utilizadas no desenvolvimento do protótipo, para realizarem, conforme dito a cima, a função de entender o comando de voz do usuário e a resposta por áudio para o mesmo.

Para Lecheta (2013, p.784):

> Essas duas funcionalidades são muito utilizadas nos aplicativos nativos do Android. Por exemplo, a capacidade de falar é utilizada pelo aplicativo GPS quando está no modo navegação, em que a voz vai falando informações sobre o trajeto. Da mesma forma, a conversão de voz em texto é muito utilizada pelos aplicativos nativos, como para disparar comandos de voz, escrever mensagens SMS por voz etc. Tudo isso ocorre graças às APIs de voz presentes no Android, seja para falar ou escutar.

Assim, com a utilização destas APIs no protótipo, cabe ao desenvolvedor montar a estrutura de entrada, para que a API possa realizar o entendimento do que foi falado, bem como a estrutura de saída, pois a partir do entendimento, deve-se realizar operações e operações para gerar a saída, ou seja, é todo um recurso estruturado.

2.9 CONSIDERAÇÕES DO CAPÍTULO

Os assuntos retratados nas seções do capítulo 2, são as pesquisas e estudos realizados para maior entendimento da tecnologia de reconhecimento de voz e também, mapeamento do processo de registro de informações em um canteiro de obras, apresentando a base bibliográfica do trabalho.

Todas estas informações auxiliam para o devido desenvolvimento de um protótipo para auxílio na inserção das medições diárias das tarefas executadas em um canteiro de obra, via comandos de voz em dispositivo móvel.

3 MÉTODO

Este capítulo aborda os meios usados para se implementar o *software* e suas delimitações, além de uma explicação mais detalhada do planejamento do desenvolvimento e detalhes do seu funcionamento.

Como descrito nos capítulos anteriores o *software* proposto neste trabalho é um aditivo para um módulo de um sistema já existente que se comunica com um servidor responsável por armazenar e prover as informações necessárias para o aplicativo.

No processo de desenvolvimento do software, será realizado a modelagem do mesmo, que é uma atividade onde constrói-se modelos de visão de negócio, visão dinâmica e visão estática do que foi proposto, para que assim, possa-se realizar o desenvolvimento, implementação e testes do software.

3.1 CARACTERIZAÇÃO DA PESQUISA

A pesquisa realizada neste trabalho é considerada e classificada como uma pesquisa aplicada. Segundo Perdigão et al (2012), é o tipo de pesquisa em que a meta é contribuir com fins práticos, buscar soluções para problemas concretos e transformar em ações concretas os resultados do trabalho.

Isso deve-se ao fato de que é preciso mais acessibilidade para o usuário que manipula dados da sua obra através de um dispositivo.

Neste, é realizada uma pesquisa sobre mobilidade no canteiro de obras por BERNSTEIN e LAQUIDARA-CARR em 2013 e segundo eles apenas 37% dos participantes da pesquisa apontaram que seus colaboradores conseguem acessar informações referentes às obras fora de suas bases, o que não somente cria possíveis problemas na produtividade, mas também limita seu potencial. A ideia deste trabalho consiste em ajudar a elevar esse índice aumentando a produtividade no canteiro de obras.

A pesquisa também pode ser classificada como bibliográfica por que foi feita a partir de materiais publicados, como livros e periódicos disponíveis na internet, entre outros, além

disso também pode-se classificar como pesquisa exploratória e bibliográfica. Assis (2012 apud. GIL, 2006, p. 43) detalha o que é pesquisa exploratória:

> Tem como finalidade proporcionar maiores informações sobre determinado assunto, facilitar a delimitação de um tema de trabalho. Normalmente constitui a primeira etapa de uma investigação mais ampla. Desenvolve-se com o objetivo de proporcionar uma visão geral, de tipo aproximativo, acerca de determinado fato.

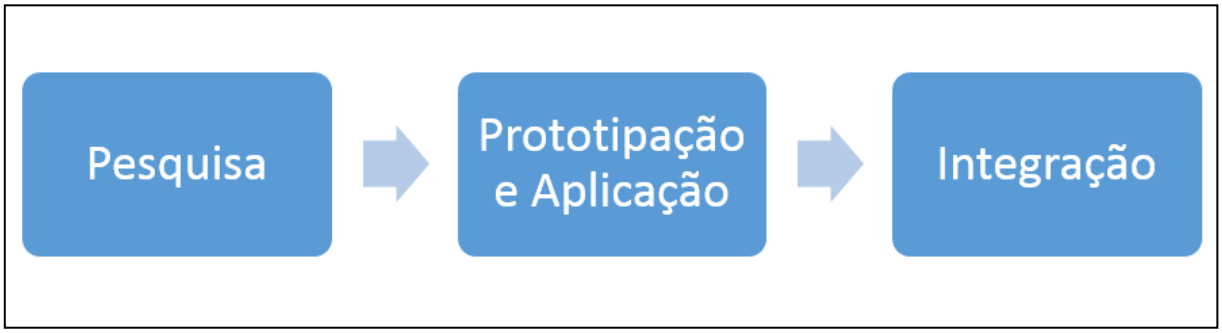
Quanto à forma de abordagem da pesquisa, ela pode ser definida como qualitativa, pois as informações não são quantificáveis já que o objetivo central do trabalho não é medir o ganho de produtividade no canteiro de obra ao utilizar o *software* proposto, os dados obtidos são analisados de forma indutiva (ASSIS, 2012).

3.2 ETAPAS METODOLÓGICAS

As etapas metodológicas foram separadas em grandes tarefas de forma que cada tarefa possa levar algumas semanas para o seu término, além disso a ordem que as tarefas são exibidas não define necessariamente a ordem de execução das mesmas, portanto, há tarefas que podem ser executadas em paralelo levando em consideração que o desenvolvimento deste trabalho é realizado por dois alunos.

As etapas são definidas em três grandes grupos sendo eles: Pesquisa, Prototipação e Aplicação e Integração, conforme apresentado na figura 10:

Figura 10: Etapas Metodológicas



Fonte: Elaborado pelo Autor

A primeira etapa é utilizada para pesquisa sobre tecnologias, bibliotecas de desenvolvimento e melhores formas de desenvolver a aplicação para não trazer problemas futuros na sua integração.

A segunda etapa consiste em utilizar o que foi pesquisado na primeira etapa para desenvolver um protótipo funcional com reconhecimento de voz ativo com a capacidade de processar o comando recebido e transformar em algum tipo de retorno que caracterize o seu correto funcionamento.

A terceira etapa é basicamente a integração do protótipo com o *Sienge* e posteriormente fazê-lo funcionar no contexto do *software* recebendo, processando e armazenando as informações relacionadas a finalidade do aplicativo móvel.

Tarefas empregadas nas etapas metodológicas:

- Pesquisar bibliotecas de reconhecimento de voz.
- Definir estratégia do processamento do reconhecimento de voz (Local ou Servidor)
- Estudar arquitetura do Sistema Operacional Android para ativação do hardware de microfone e Rede.
- Estudar arquitetura aplicada no *Sienge Mobile* para definir melhor caminho à seguir no desenvolvimento do *software* de processamento e reconhecimento de voz.
- Desenvolver protótipo de *software* em Android para aprender a utilizar as ferramentas de desenvolvimento disponíveis.
- Utilizar as bibliotecas de reconhecimento de voz pesquisadas e realizar testes de desempenho e facilidade de uso (programação) em um dispositivo móvel, definir prós e contras e escolher uma para utilizar na aplicação.
- Instalar o Sienge Mobile no dispositivo e obter o código fonte e configurar a IDE com o mesmo.
- Integrar o protótipo ao Sienge Mobile de forma que seja possível acessar o protótipo criado anteriormente.
- Aprimorar o protótipo para que ele funcione em conjunto com a aplicação fornecendo e armazenando informações obtidas através do reconhecimento de voz.

- Realizar testes com dispositivos de microfone Bluetooth.

3.3 PROPOSTA DA SOLUÇÃO

A solução proposta é baseada na tecnologia Android e é um aditivo para o módulo de registro de medição física de mão de obra, na figura 11 representa o processo de comunicação do *software*:

Figura 11 - Diagrama do esquema tecnológico

Fonte: Elaborado pelo autor

A figura 11 mostra como o *software* de reconhecimento de voz será integrado a plataforma já existente, basicamente ele será apenas uma segunda forma de inserir dados no sistema, ele se utilizará das funções do sistema para realizar as operações garantindo assim mais integralidade e produtividade levando em consideração que caso haja uma atualização no sistema só haverá um único ponto para ser alterado.

Para fins de configurações, segurança e demais operações que o *software* necessite que não seja direta ou indiretamente ligada ao módulo de registro de medição física da obra é realizada através da tecnologia *REST* para comunicação direta com o servidor ERP.

REST é um estilo arquitetural para sistemas hipermídia distribuídos, que enfatiza a generalização das interfaces, a escalabilidade da interação entre os componentes e a instalação independente dos mesmos (FIELDING, 2000).

Com a tecnologia *REST* a aplicação pode solicitar serviços para o ERP Sienge sem acoplamento de forma que não seja preciso alterar coisas que já existem para suportar esse novo cliente neste caso o *software* de reconhecimento de voz.

3.4 DELIMITAÇÕES

Devido ao tempo limitado para construção do trabalho, não haverá um protótipo funcional para dispositivos que não rodam o sistema operacional Android, como por exemplo iOS da *Apple Inc.*

O funcionamento da aplicação com microfones de terceiros pode falhar em alguns momentos além do próprio microfone do dispositivo móvel. O funcionamento do serviço de reconhecimento de voz em plataformas inferiores ao *Android 4.0 Ice Cream Sandwich* pode gerar inconsistências.

A modelagem do *software* será em baixo nível enfatizando as regras e requisitos do projeto, assim como os casos de uso que irá suportar, em outras palavras não serão desenvolvidos diagramas estruturais ou de iteração, apenas comportamentais que visam detalhar e documentar cenários de experiência do usuário.

4 MODELAGEM

Este capítulo apresenta a modelagem do desenvolvimento de aplicação para dispositivos móveis proposta neste trabalho. Esta tem o propósito de melhorar o processo de inserção das informações durante os trabalhos de medições de obras em uma construção civil.

A modelagem, a qual é uma atividade onde realiza-se a elaboração de modelos e padrões definidos para o desenvolvimento, segundo Pereira (2013, p.4):

> É uma linguagem gráfica para visualizar, especificar, construir e documentar os artefatos de um sistema computacional orientado a objetos. Trazendo vantagens, como: Desenvolvimento de programas de forma rápida, eficiente e efetiva, revela a estrutura desejada e o comportamento do sistema, permite a visualização e controle da arquitetura do sistema e melhor entendimento do sistema que está sendo construído e gerenciamento de riscos (PEREIRA, 2013, p.4).

Estes são mapeamentos gráficos e escritos para auxílio na etapa de desenvolvimento de uma aplicação.

4.1 PROPOSTA DA SOLUÇÃO

Um protótipo com a função de tentar agilizar o processo de mapeamento de obras na construção civil é aqui apresentado. Este busca minimizar o índice de erro na absorção dos dados durante o processo de aquisição de dados. Em tempo, o processo de aquisição de dados deve ser efetuado por comandos de voz como uma forma mais intuitiva e ágil.

4.2 CONSTRUÇÃO DO MODELO

Assim como em qualquer novo projeto ou software a ser criado, tem-se um modelo prévio para servir de base ou de uma representação daquilo que será construído. Conforme apresentado pelo Lobo (2008, p.55):

> Na criação de produtos são utilizados modelos, normalmente com o uso de símbolos e desenhos técnicos, que tem o objetivo de representar todas as suas características relevantes. Em cada engenharia, existe uma forma técnica de utilizar símbolos e diagramas para criar modelos da peça, do produto ou do sistema a ser construído. Na modelagem de software não seria diferente, através de diagramas são construídos modelos que tem a função de representar o software de forma objetiva (LOBO, 2008, p55).

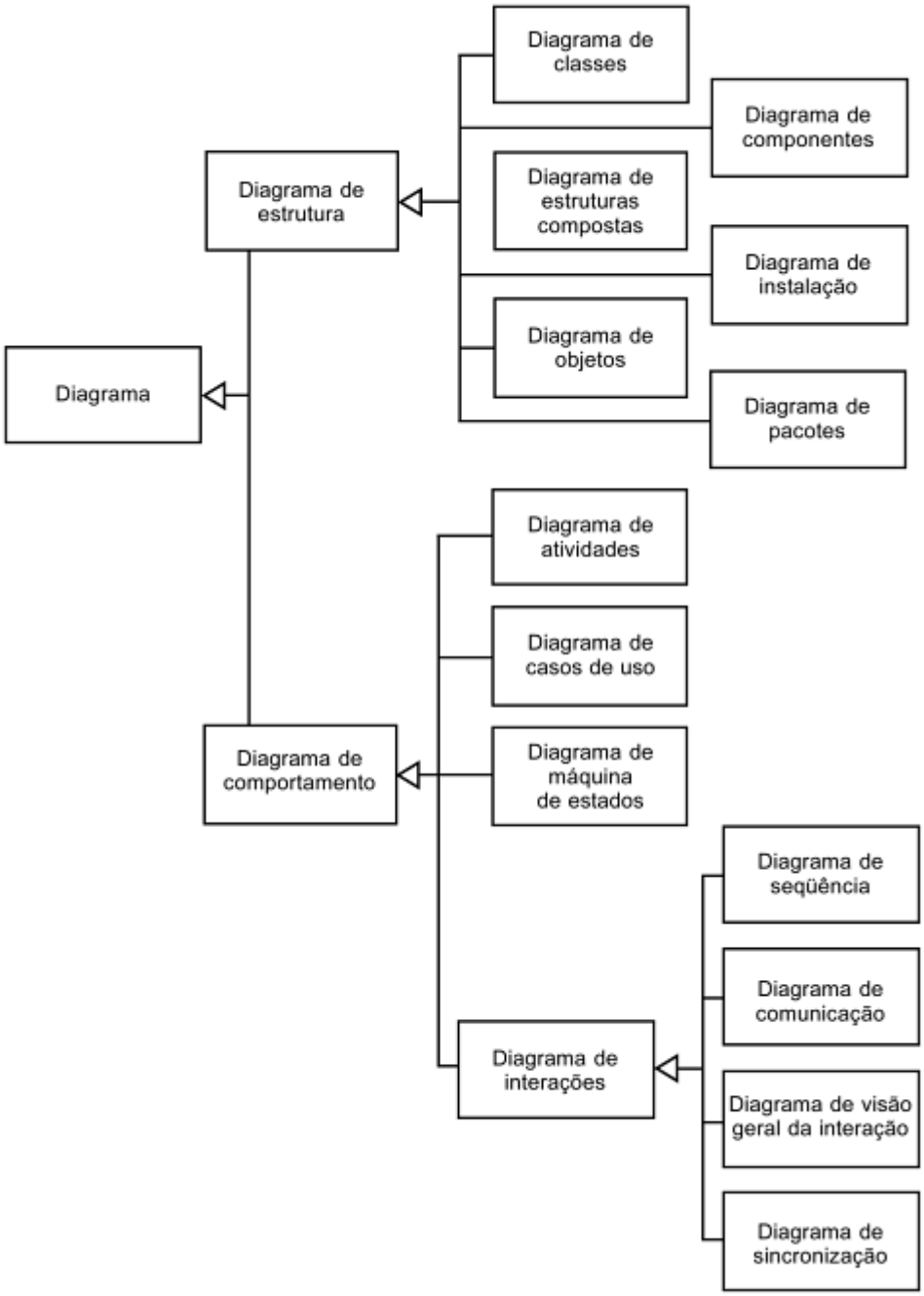
Em desenvolvimento de softwares, a UML (*Unified Modeling Language*) é bastante utilizada para poder transmitir alguns aspectos de um sistema. UML é um conjunto de notações gráficas desenhadas em um software específico, que servem para representar um esboço, projeto e linguagem de programação (FOWLER, 2004, p. 25).

Para Fowler (2004, p. 26):

> A essência dos esboços é a seletividade. No esboço para desenvolvimento, você delineia alguns problemas em código que você está prestes a escrever, normalmente discutindo-os com um grupo de pessoas de sua equipe. Seu objetivo é usar os esboços para ajudar a transmitir as ideias e alternativas sobre o que está prestes a fazer. [...] UML como projeto tem como foco a completeza. No desenvolvimento, a ideia é de que os projetos são desenvolvidos por um projetista, cujo trabalho é construir um projeto detalhado para um programador codificar. Esse projeto deve ser suficientemente completo, no sentido de que todas as decisões estejam expostas, e o programador deve ser capaz de segui-lo (FOWLER, 2004, p.26).

A UML possui seus diagramas para que possa ser representado todo este esboço e projeto para os desenvolvedores, para que assim, possam, a partir do que foi desenhado e modelado, partir para a escrita do software, sempre seguindo os diagramas. Estes diagramas estão listados e classificados conforme indicado na figura 12:

Figura 12: Classificação dos tipos de diagrama da UML



Fonte: (FOWLER, 2004, p. 34)

Cada bloco desta estrutura é uma representação gráfica do software, que auxilia na visualização do desenho, ou seja, demonstração em diagramas padronizados, especifica significados para o código e demonstra a comunicação entre os objetos relacionados.

4.2.1 Requisitos

Requisitos fazem parte de qualquer projeto. Estes abrangem as primeiras visões, as funcionalidades e as restrições daquele sistema ou projeto.

Estes, são as características do sistema e se dividem em requisitos funcionais e não funcionais. Os requisitos funcionais definem os caminhos que o sistema possui, ou seja, as funcionalidades desenvolvidas e quais manipulações das informações podem ser realizadas. Já os requisitos não funcionais definem as plataformas com as quais o sistema pode ser utilizado, termos de conectividade entre sistema, servidor e acessórios, performance e os meios com os quais o sistema interage com o usuário final. (MARTINS, 2007, p. 208)

Assim, quando se tem uma especificação desta modelagem de requisitos, tanto funcionais, quanto não funcionais, é, atualmente, considerada uma abordagem extremamente adequada, pois com isto, facilita a comunicação entre equipe de projeto e os clientes/usuários, e, ainda promove a comunicação, o gerenciamento e a condução do desenvolvimento do projeto negociado (RAMOS, 2006).

Complementando, segundo Martins (2007, p. 208):

> O objetivo deste processo é definir as características do sistema conforme observado pelo cliente, apontando o desenvolvimento na direção correta. O primeiro passo é capturar os requisitos apresentados pelos stakeholders (usuários, clientes, marketing, etc.) composto por solicitações e desejos. O resultado será um documento com a Visão do Sistema, que apresenta as suas características técnicas e funcionais. Posteriormente, estes requisitos são traduzidos em requisitos detalhados, através das ferramentas do UML, de modo que possa ser criada uma arquitetura para o sistema.

Com todo o descritivo a cima sobre os requisitos de um sistema, abaixo segue a modelagem dos requisitos funcionais e não funcionais capturados deste protótipo.

4.2.1.1 Requisitos funcionais

A seguir são apresentados os Requisitos Funcionais:

Quadro 2 – Requisitos Funcionais

Requisitos Funcionais	
Código Requisito	**Descrição**
RF001	Deve ser possível solicitar a locução das obras disponíveis para realizar medições
RF002	Deve ser possível solicitar a locução das unidades construtivas de cada obra
RF003	Deve ser possível solicitar a locução das tarefas e seus respectivos agrupadores de cada unidade construtiva
RF004	Deve ser possível solicitar a locução das informações de cada tarefa
RF005	Deve ser possível alterar a quantidade de trabalho realizado de cada tarefa individualmente
RF006	Deve ser possível fechar o aplicativo através de comando de voz

Fonte: Elaborado pelo Autor

4.2.1.2 Requisitos não funcionais

A seguir é apresentado os Requisitos Não Funcionais:

Quadro 3 – Requisitos Não Funcionais

Requisitos Não Funcionais	
Código Requisito	**Descrição**
RNF001	O sistema deve operar sem a conexão de rede: Caso ocorra de não possuir rede ou de não ter rede, o sistema deve continuar em operação
RNF002	O sistema deve rodar em plataforma Android igual ou superior a 4.0
RNF003	O sistema deve funcionar com acessórios externos (Fones de ouvido, microfones)
RNF004	Deve ser possível controlar todo o sistema por voz
RNF005	O sistema deve responder o usuário através de áudio
RNF006	O sistema deve possuir a capacidade de informar os comandos de voz disponíveis dentro de cada contexto

Fonte: Elaborado pelo Autor

Conforme representado nos quadros a cima, o protótipo deve ter estes requisitos quando estiver em execução, ou seja, caso ocorra o que foi determinado no requisito funcional ou não funcional, o sistema deve agir conforme foi descrito.

4.3 DIAGRAMA DE CASO DE USO

O diagrama de caso de uso demonstra a interação que ocorre entre os atores e os casos de usos que o sistema tem, demonstrando a ligação que os artefatos têm entre si.

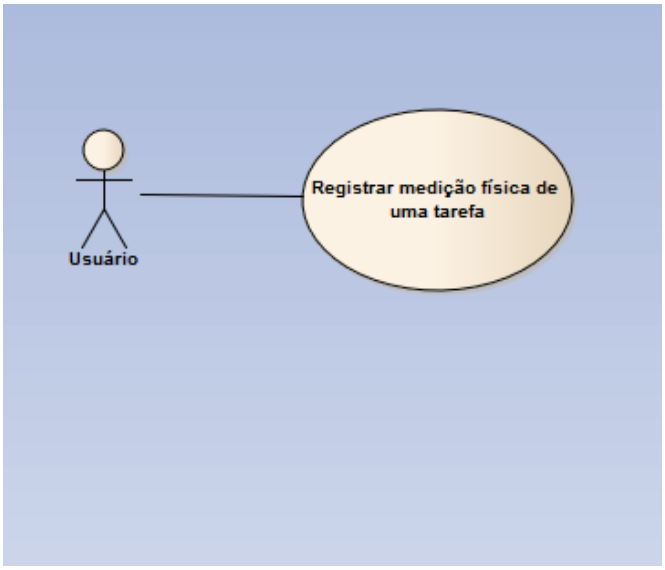
Segundo Booch et al (2005, p.243), “um diagrama de caso de uso é um diagrama que mostra um conjunto de casos de uso e atores e seus relacionamentos”.

Para Ramos (2006, p. 65):

> O modelo de casos de uso permite capturar os requisitos de um sistema por meio do detalhe de todos os cenários que os usuários podem acessar. Os casos de uso, mais do que iniciar a modelagem de requisitos de um sistema, conduzem todo o processo de desenvolvimento. [...] Um caso de uso deve descrever o que faz um sistema (ou parte dele), e não como ele é realizado. O foco é, portanto, na visão externa do sistema, ou seja, na visão que os usuários têm dele.

Assim sendo, o principal objetivo do sistema é facilitar a inclusão de registros de medição física de uma obra e este em um primeiro momento é o único caso de uso do sistema, representado na figura 13.

Figura 13: Diagrama de Caso de Uso: Registro de medição física de uma obra



Fonte: Elaborada pelo autor

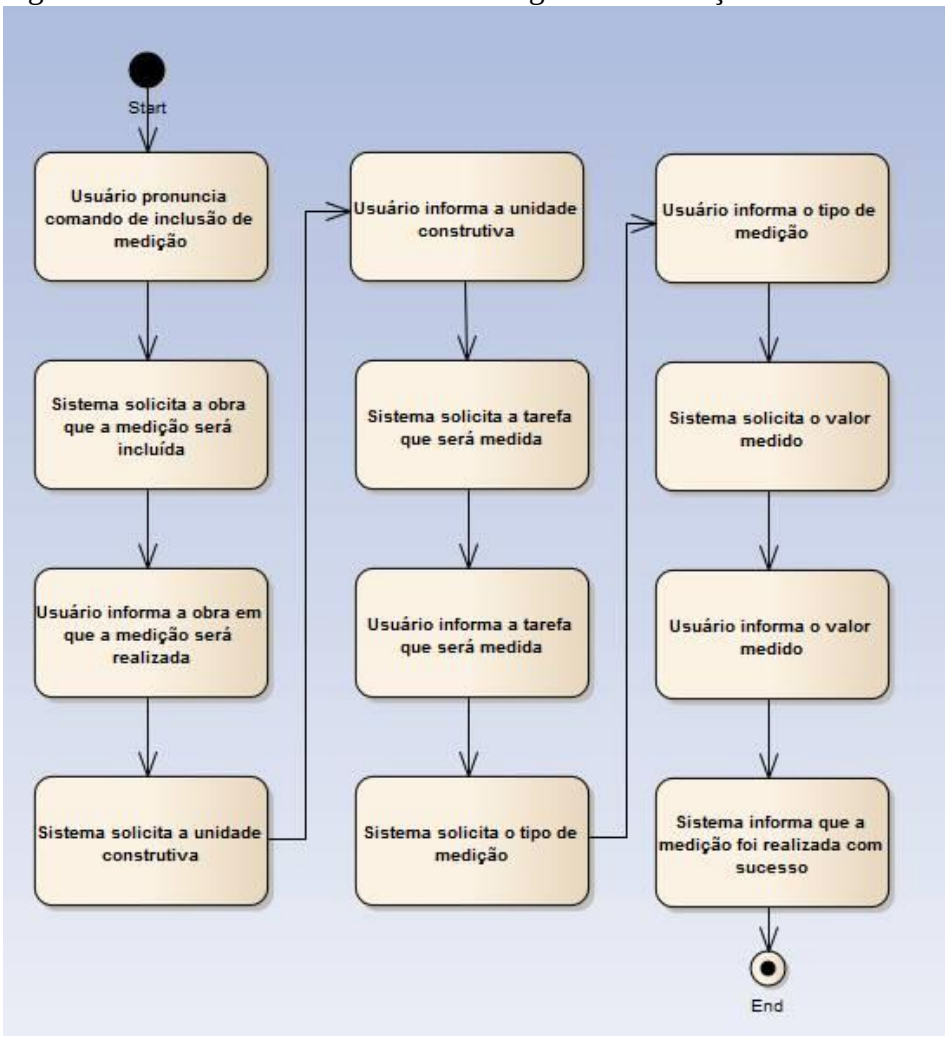
O fluxo do cenário básico do caso de uso a cima, é representado na figura 14. Este fluxo é longo pois como o usuário está conversando com a aplicação, ele necessita de feedbacks constantes para ter segurança de que as operações que está solicitando através da

fala estão sendo interpretadas corretamente pela aplicação, por este motivo, o registro da medição é feito em partes e a cada passo finalizado o sistema informa ao usuário o sucesso ou não no passo executado.

O sistema deve sempre manter o usuário informado sobre o que está acontecendo por meio de feedback apropriado dentro de um tempo razoável (ANDRADE, 2007, p. 57);

Na figura 14, é demonstrado este fluxo de cenário básico do caso de uso registro de medição física de uma obra:

Figura 14 – Fluxo do Cenário Básico: Registro de medição física de uma obra



Fonte: Elaborada pelo autor

Complementando a figura 14, a cada operação realizada, seleção da obra, seleção da unidade construtiva, seleção da tarefa o sistema informa o usuário que o termo foi entendido e associado corretamente com uma informação presente na aplicação e só quando correto solicita a próxima informação até chegar ao fim do processo.

4.4 DIAGRAMA DE CLASSE

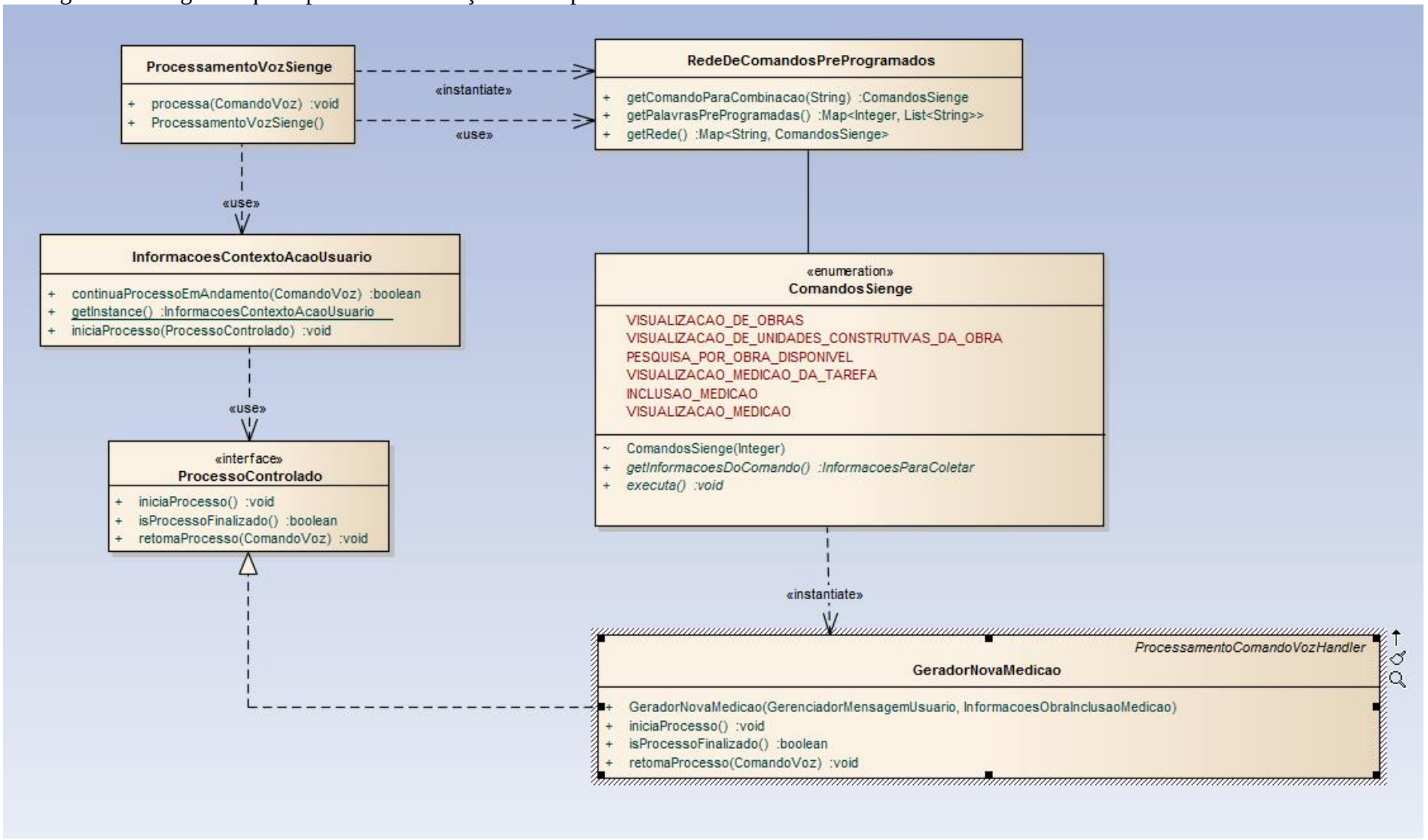
Dentro do software existe um conceito chamado processo controlado, é um processo no qual o sistema precisa de várias informações que o usuário precisa informar, mas não pode obter todas elas ao mesmo tempo, o sistema então por sua vez, a cada iteração realiza uma fase nova do processo solicitando uma informação diferente em outras palavras, é a abstração de um fluxo de operações pré-determinadas em que o software solicita informações do usuário durante o processamento de uma ação.

Estes processos controlados são criados individualmente dentro do software para cada ação controlada existente, uma ação se torna um processo controlado quando implementa a interface Processo Controlado, este processo precisa ser iniciado através de um gerenciador de contextos que é notificado sempre que um novo processo controlado é iniciado pelo usuário, a estrutura de um Processo Controlado é simples, ele é iniciado através do método *iniciaProcesso*, quando o usuário entra com uma nova informação ele é retomado através do método *remotaProcesso* recebendo as informações do usuário e é finalizado pelo gerenciador de contextos quando o método *isProcessoFinalizado* retorna *true* para o gerenciador de contextos.

O principal processo controlado da aplicação é a geração de uma nova medição, em que o sistema solicita em partes ao usuário as informações necessárias para se incluir uma medição. A figura 15 representa um diagrama de classes de um processo controlado utilizando como exemplo a geração de uma nova medição.

Segundo Sampaio (2007, p. 23) “um Diagrama de Classe é uma visão estática de um modelo de Objetos. Como um modelo relacional (Entidade Relacionamento) ele descreve as Entidades, neste caso, Classes, e os relacionamentos entre elas”.

Figura 15: Diagrama que representa a execução de um processo controlado.



Fonte: Elaborada pelo Autor

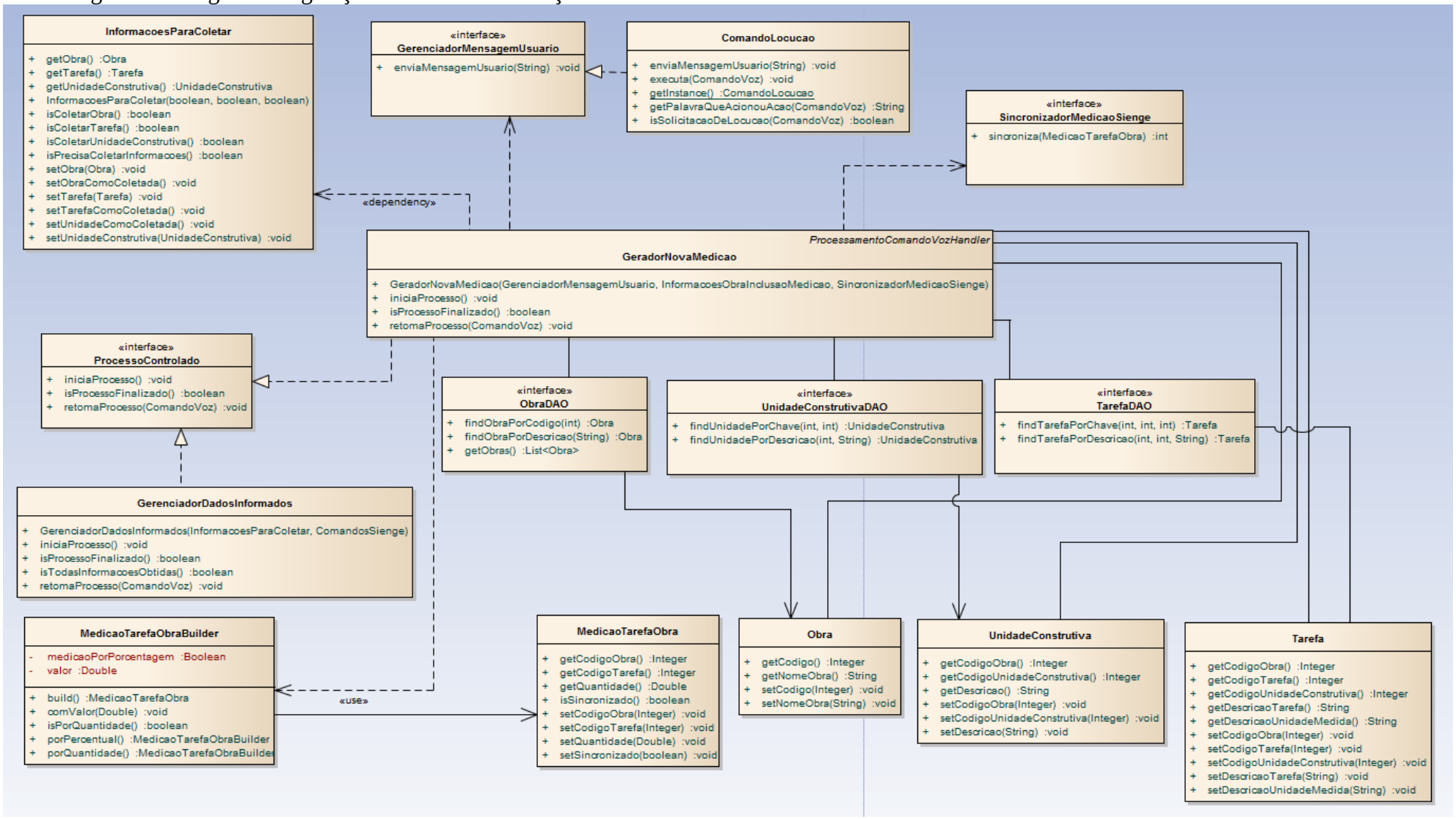
A geração de uma nova medição é a base principal do sistema, envolvendo muitas operações no contexto da inclusão por voz. Apesar de ser um processo controlado e possuir uma interface simples para quem o executa, possui diversos artefatos necessários em sua composição, além da necessidade de realizar a validação e conferência dos dados informados pelo usuário, possui também, a responsabilidade de montar uma nova medição com estes dados e, posteriormente, sincronizar com o ERP. Todas estas tarefas fazem parte deste único processo controlado.

Para a construção da medição utilizou-se do padrão de projeto *Builder*, Gamma e outros (2009, p. 24) descrevem a intenção do padrão Builder como, "Separa a construção de um objeto complexo da sua representação, de modo que o mesmo processo de construção possa criar diferentes representações ".

Já para Kerievsky (2008, p.126) "Embora criar diferentes representações de um objeto complexo seja um serviço útil, não é o único serviço que um *Builder* provê. Simplificar a construção ou desacoplar código cliente de um objeto complexo são também razões igualmente boas para usar um *Builder"*.

O diagrama de classes da geração de uma nova medição pode ser visto a seguir na figura 16:

Figura 16: Diagrama de geração de uma nova medição.



Fonte: Elaborada pelo Autor

Cada diagrama aqui apresentado foi elaborado a partir dos requisitos relacionados, podendo assim, extrair os objetos da aplicação e modelá-los em um desenho gráfico, onde cada bloco mostra seu relacionamento com o outro bloco e o seu tipo de interação, conforme mostrado nas figuras a cima, figura 15 e 16 (MELO, 2010, p.97).

4.5 DIAGRAMA DE DEPLOYMENT

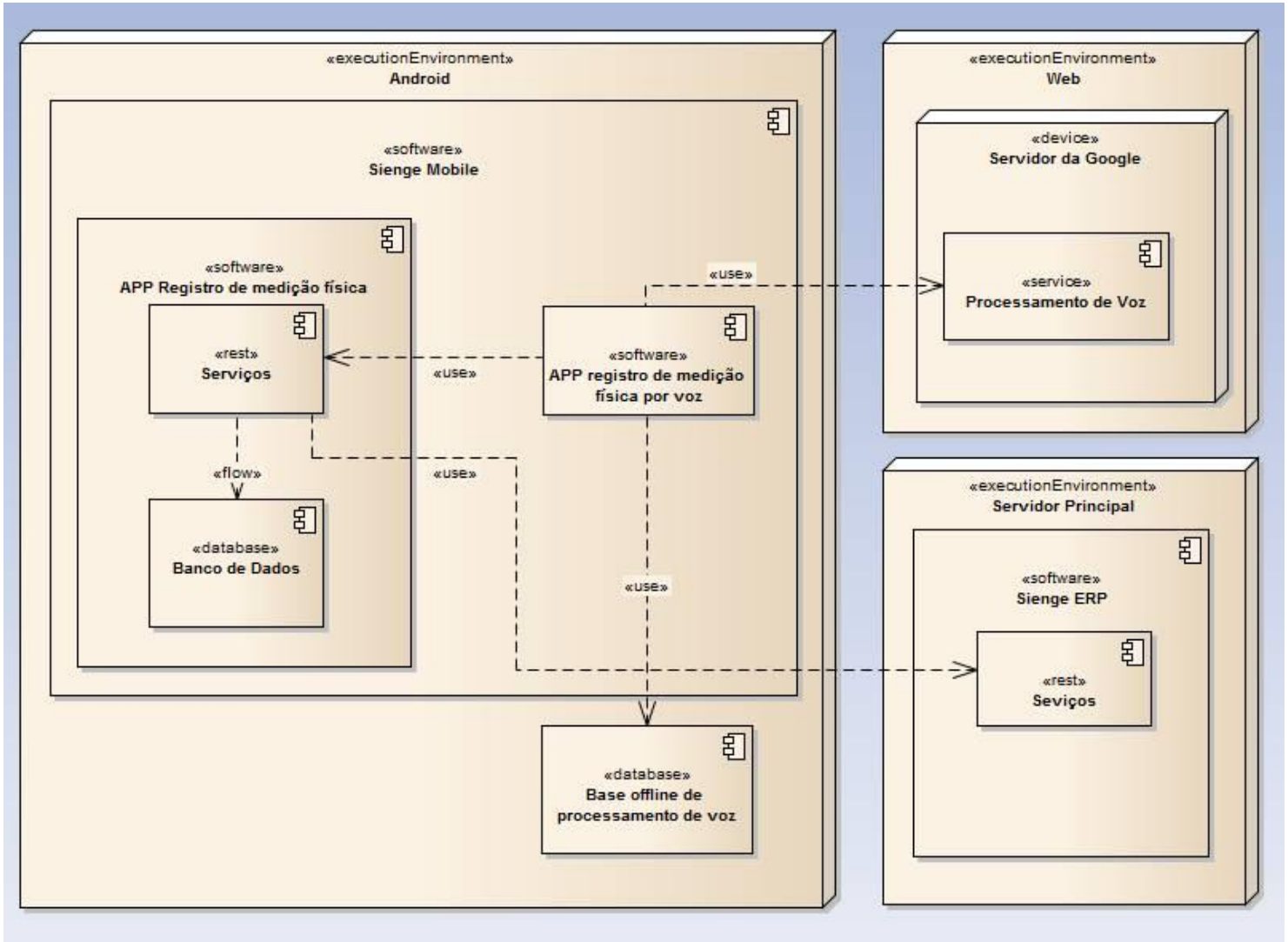
Para representação da estrutura física da implementação do sistema, demonstrando todos os componentes envolvidos, bem como seus relacionamentos e interfaces é usado o Modelo de Implantação, podendo ser chamado também de Diagrama de Deployment. Para Martins (2007, p. 9) "é usado como parte do projeto da arquitetura para mostrar os subsistemas e o ambiente computacional onde eles serão alojados, incluindo o hardware e o sistema operacional que serão utilizados, mais a localização dos pacotes de componentes de software". Graficamente, este diagrama contém nós e os relacionamentos entre si.

Complementando por Rezende (2005, p. 212):

> Mostra a estrutura do sistema com suas relações físicas entre os componentes de software e configurações de hardware a serem implantados (run-time). Mostra como os componentes e objeto são acessados e se movem num sistema distribuído. Trata-se de um gráfico de nós conectados por associações de comunicação, que podem conter instâncias de componentes, classes, bibliotecas ou executáveis, inclusive os desenhos dos sistemas de telecomunicações e os sistemas operacionais

A representação do diagrama de deployment do protótipo pode ser visto a seguir na figura 17:

Figura 17: Diagrama de Deployment do Protótipo



Fonte: Elaborada pelo Autor

Conforme representação na figura 17 do diagrama de deployment, o protótipo inicia através da tarefa de registro de medição física, por meio de um botão em uma tela. Após iniciada a aplicação, a mesma torna-se auto gerenciável, quando houver necessidade, consumindo serviços *REST* disponíveis no aplicativo principal.

Um exemplo disto, é quando o sistema se encontra *off-line* e as consultas ou as inserções dos dados das operações utilizam um banco de dados interno da aplicação, onde essas informações sincronizam-se com o ERP Sienge posteriormente, tornando a aplicação por voz apenas uma forma de entrada de dados, diferente para a tarefa de registro de medição no modo *online*, que por sua vez, consome os serviços de reconhecimento e processamento da voz dos servidores da Google para refinar a identificação.

4.6 DESCRIÇÃO DO PROTÓTIPO

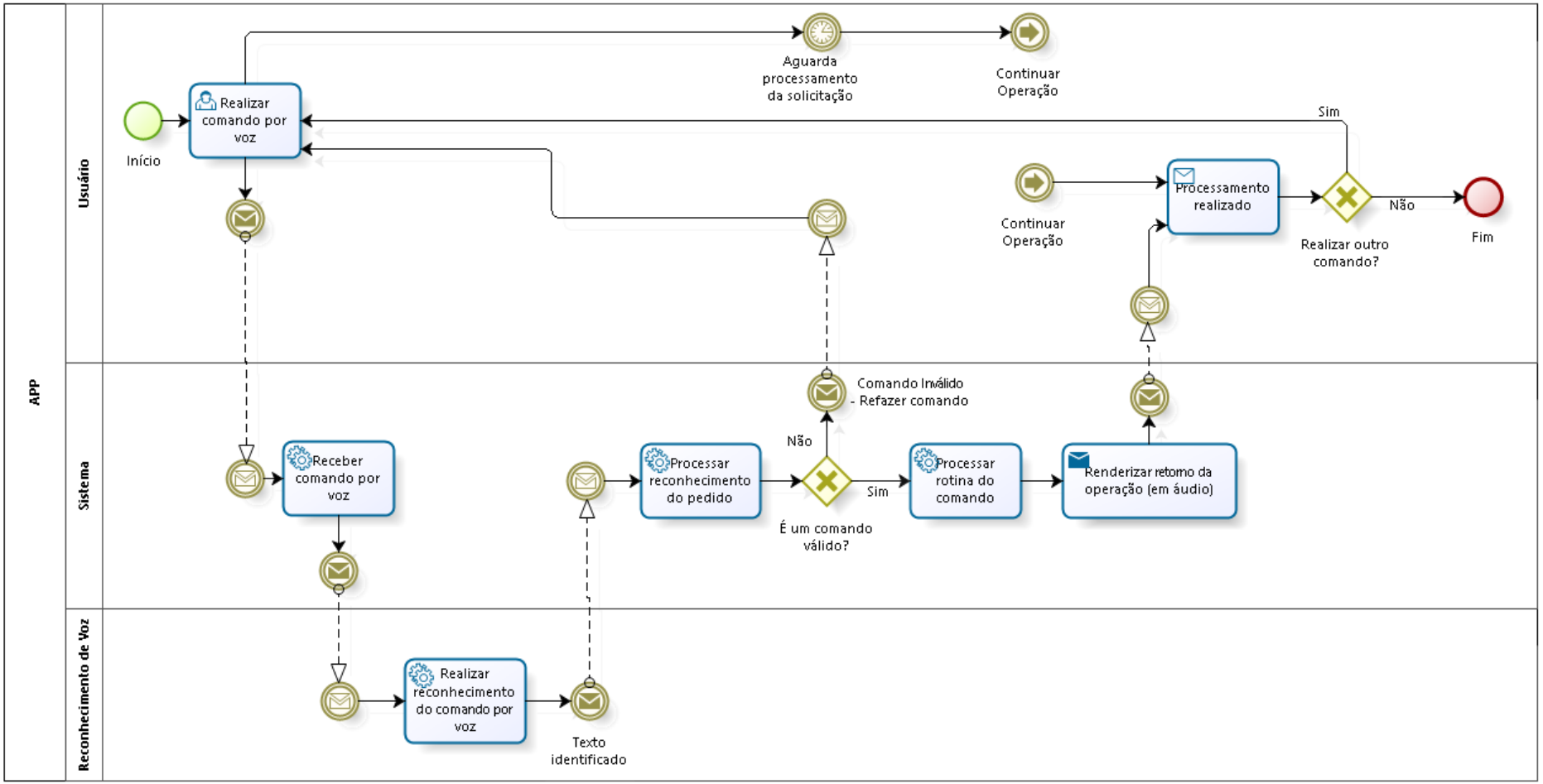
O sistema consegue identificar as solicitações do usuário quando não está conectado na internet, porém, com uma perda de qualidade na interpretação, ou até não realizando a identificação do que foi falado, além disso o tempo de processamento fica um pouco elevado quando comparado ao tempo de processamento obtido quando está *online*. O motivo disto é que quando o sistema se encontra *off-line* utiliza-se de um banco de dados interno e limitado para identificar o texto recebido, palavras ou termos que não se encontram no dicionário, como por exemplo as gírias ou preposições não utilizadas comumente ocorrendo a uma interpretação incorreta por parte do sistema. Já quando encontra-se conectando na internet, o programa utiliza os serviços do Google para realizar as interpretações por meio de mecanismos de pesquisa para identificar as palavras mais utilizadas dentro do contexto do texto interpretado, em outras palavras, o reconhecimento de voz torna-se muito mais poderoso e eficiente quando está conectado à internet. Para que o mesmo funcione de forma satisfatória, quando não conectado à internet, exige um custo alto de desenvolvimento para tornar o programa mais inteligente, a nível que identifique as palavras corretas dentro do contexto atual, priorizando assim, as palavras mais comuns naquele contexto.

A aplicação inicia em background aguardando solicitações através de comandos de voz do usuário, quando o sistema recebe um comando redireciona este para um sistema de

reconhecimento de voz que por sua vez processa este comando ordenando por probabilidade de acerto daquilo que foi falado e o retorna em formato de texto. O texto retornado é processado pelo sistema que tenta realizar a identificação deste comando para posteriormente executar uma rotina especifica, se o comando não for identificado o sistema retorna uma mensagem de voz para o usuário informando-o que o comando não foi compreendido, caso o sistema entenda o comando, a rotina é executada e o sistema informa o usuário através de voz que o comando foi executado com sucesso, permitindo que ele realize uma nova solicitação.

Diagrama do sistema é apresentado na figura a seguir.

Figura 18 – Diagrama do funcionamento do sistema e interação com o usuário.



Fonte: Elaborada pelo Autor

O diagrama representado na figura 18, mostra as atividades e o caminho que o protótipo deve seguir, a partir das solicitações realizadas pelo usuário. Foi representado no modelo de BPMN[26], demonstrando o fluxo de atividades, bem como as configurações que cada uma exerce dentro do sistema.

[26] BPMN: Notação de modelagem de processos de negócio

5 DESENVOLVIMENTO

Descrevem-se neste capítulo o histórico do desenvolvimento, que vai desde o desenvolvimento em si, até um detalhamento dos problemas encontrados na implantação bem como as soluções tomadas e os métodos utilizados para a validação do sistema.

5.1 ESCOLHA DAS TECNOLOGIAS

A tecnologia escolhida para o reconhecimento e a síntese de voz teria que atender os requisitos não funcionais que se aplicam a este tema, e que foram especificados pela Softplan disponíveis no capítulo 4, além de possuir bastante documentação disponível para facilitar o desenvolvimento. Após pesquisar algumas APIs disponíveis no mercado escolhemos utilizar as tecnologias nativas do Android para o desenvolvimento da aplicação pois a mesma atendia requisitos e tinha o bônus de ser compatível com qualquer celular Android.

5.2 UTILIZAÇÃO DA API TEXT TO VOICE

A API de fala do Android é muito simples de utilizar, porém, para que ela funcione corretamente é necessário configurar uma voz na linguagem desejada nas configurações do dispositivo normalmente rotulada pelo texto “Idioma e inserção”, dentro das configurações de “Idioma e inserção” deve existir algum menu com algum texto semelhante a “Opções de texto-para-fala”, dentro dessa funcionalidade é instalar novas vozes e realizar algumas configurações para a voz instalada.

Isso permite que cada usuário configure a voz que mais gosta para conversar consigo durante o ciclo de funcionamento da aplicação.

Porém, antes de sair utilizando a função principal da API que é transformar texto em voz é necessário se certificar se os pacotes de voz estão instalados e configurados no

dispositivo, ou seja, se o TTS está funcionando, se ele não estiver instalado a própria aplicação vai tentar iniciar o processo de instalação para que o aplicativo funcione como demonstrado no código abaixo.

Código 1: Código para verificação ou instalação do TTS

```
private void verificaSeFuncionalidadeTextToSpeechEstaAtiva() {
	PackageManager pm = contextoPrincipal.getPackageManager();
	List<ResolveInfo> activities;
	Intent intentCheckTTS = newIntent(TextToSpeech.Engine.ACTION_CHECK_TTS_DATA)
	activities = pm.queryIntentActivities(intentCheckTTS, 0);

	if (activities.size() == 0) {
		instalaTTS();
	}
}


private void instalaTTS() {
	Intent installIntent = new Intent();
	installIntent.setAction(TextToSpeech.Engine.ACTION_INSTALL_TTS_DATA);
	contextoPrincipal.startActivity(installIntent);
}
```

Realizada a verificação no TTS, conforme demonstrado no código 1, para utilizar o TTS precisa-se de uma classe que implemente a interface *OnInitListener* do pacote *android.speech.tts.TextToSpeech* essa interface obriga a implementação do método *onInit* que é executado quando a inicialização do TTS termina, independentemente de ter sido inicializada com sucesso ou não, caso sua inicialização não ter sido com sucesso é abortada a continuação no processo.

Código 2: Implementação do onInit para utilização do TTS

```
@Override
public void onInit(int status) {
	if(status == TextToSpeech.SUCCESS) {
		locutorInicializado = true;
	}
}
```

Enquanto a variável *locutorInicializado* que é um atributo de classe da classe que implementa a interface *onInitListener* não for configurado com *true*, o sistema não estará pronto para realizar locuções, conforme código 2.

Após essas verificações é necessário apenas uma linha de código para que a aplicação realize locuções, conforme demonstrado código 3.

Código 3: Código para locução de texto

```
public void fala(String texto) {
        speaker.speak(texto, TextToSpeech.QUEUE_FLUSH, null);
        while(speaker.isSpeaking()) {
                aguardaUmSegundo();
        }
}
```

A chamada deste método é composta por três parâmetros, o primeiro que é o texto à ser falado pelo Android, o segundo é uma configuração, que segundo a descrição disponível na documentação da API (http://developer.android.com/reference/android/speech/tts/TextToSpeech.html) tem duas opções disponíveis *QUEUE_FLUSH* ou *QUEUE ADD*, a primeira configura o locutor para falar exatamente o que está escrito no parâmetro texto e a segunda configuração configura-o de forma que o parâmetro texto seja acrescentado ao final da última locução realizada e que todo o bloco de texto (antigo e novo) seja lido inteiro.

A variável speaker é um atributo de classe e é a classe que controla a locução de texto no Android, ela foi configurada no construtor da classe como demonstrado no código 4:

Código 4: Código de instanciação da API TTS

```
speaker = new TextToSpeech(contextoPrincipal, this);
speaker.setLanguage(new Locale("pt-br"));
speaker.setOnUtteranceProgressListener(controleLocucao);
```

Para instanciar um locutor, que no caso da aplicação se chama speaker, é necessário passar para o construtor uma instância da *activity* principal, chamada contexto principal. Assim sendo, foi configurado o idioma a ser utilizado no método *setLanguage()*, que é o parâmetro que o Android usa para selecionar a voz no idioma certo, a terceira linha é a injeção de uma dependência da interface *UtteranceProgressListener*, ela fornece alguns

métodos de controle, como por exemplo: executar uma operação quando o locutor terminou de pronunciar o texto enviado.

5.3 UTILIZAÇÃO DA API SPEECH TO TEXT

A API de reconhecimento de voz do Android possui muitos recursos, o que torna a sua utilização interessante além de ter um funcionamento simples.

Assim como a API de fala do Android precisa verificar se os pacotes de voz estão instalados, a API de reconhecimento precisa realizar uma operação semelhante como pode ser visto no código 5.

Código 5: Código para verificação do reconhecimento de voz

```
public void verificaSeReconhecimentoDeVozEstaAtivo() {
        PackageManager pm = getPackageManager();
        Intent intentCheckRecognizer = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
        List<ResolveInfo> activities = pm.queryIntentActivities(intentCheckRecognizer, 0);
        if (activities.size() == 0) {
                informaUsuarioQueOReconhecimentoNaoEstaConfigurado();
        }
}
```

Assim como o TTS, o reconhecimento de voz precisa ser configurado previamente no dispositivo, através das configurações de “Idioma e inserção”, na seção “Reconhecedor de voz”, neste caso os dispositivos precisam estar configurados com o reconhecedor de voz do Google.

Durante testes nas primeiras versões do protótipo, foi constatado que a interface de reconhecimento do Google, composta por um ícone de microfone e um texto, sobrepunha a aplicação. Isso prejudicava a visualização da interface do usuário do aplicativo, dessa forma, não foi utilizada a forma habitual da API de reconhecimento de voz do Google, mas uma maneira de efetuar esse procedimento por meio de um serviço.

Para iniciar a aplicação como um serviço precisa-se ter uma classe filha da classe *Service* do pacote *android.app.Service,* esta classe que agora é um serviço precisa ser

configurada como um serviço no arquivo de configuração da aplicação, chamado *AndroidManifest.xml* que pode ser visto no código 6:

Código 6: Arquivo de configuração da aplicação

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.sienge.assistente_sienge"
   android:versionCode="1"
   android:versionName="1.0" >

   <uses-sdk
      android:minSdkVersion="16"
      android:targetSdkVersion="16" />

   <uses-permission android:name="android.permission.INTERNET" />
   <uses-permission android:name="android.permission.RECORD_AUDIO" />

   <application
      android:allowBackup="true"
      android:icon="@drawable/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
      <activity
         android:name="com.sienge.assistente_sienge.AssistenteSienge"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
      <service  android:name="com.sienge.reconhecimento.ativacao.AtivacaoReconhecimentoVozService"/>
   </application>

</manifest>
```

A classe filha *de android.app.Service* é na aplicação é chamada de AtivacaoReconhecimentoVozService e é configurada no arquivo através do seletor <service/>.

Além disso é necessário também as permissões de INTERNET e RECORD_AUDIO que permitem que o reconhecimento de voz da Google ouça o microfone do celular e se conecte com o servidor para aprimorar o reconhecimento.

A iniciação da detecção de voz deve ser iniciada pelo serviço quando este estiver em execução, para iniciar o reconhecimento de voz é necessário criar uma *Intent* configurada com esse objetivo, essa *Intent* pode possuir algumas configurações específicas do reconhecimento

de voz que podem ser moldadas com base na necessidade da aplicação como demonstrado abaixo.

Código 7: Intent reconhecimento de voz

```
public static Intent criaIntentReconhecimentoVoz() {
        Intent intent = new Intent(ACTION_RECOGNIZE_SPEECH);
        intent.putExtra(EXTRA_LANGUAGE_MODEL, LANNGUAGE_MODEL_WEB_SEARCH);
        intent.putExtra(EXTRA_PARTIAL_RESULTS, true);
        intent.putExtra(EXTRA_SPEECH_INPUT_POSSIBLY_COMPLETE_SILENCE_LENGTH_MILLIS,
                        3000l);
        intent.putExtra(EXTRA_SECURE, true);
        return intent;
}
```

A *intent* foi configurada de forma que quando a Internet estiver disponível o reconhecimento será feito utilizando a web para aprimorar o resultado, que a aplicação traz mais de um resultado para o que foi entendido de forma que seja possível escolher o comando mais correto, também foi configurado que o reconhecedor aguarde até 3 segundos de silêncio antes de parar o reconhecimento, isso serve para que quando um usuário fale pausadamente, o tempo entre as palavras faladas não interrompa o reconhecimento, todas as constantes demonstradas no código 7 são originadas da classe *RecognizerIntent.*

Depois de criar a *intent* é necessário executá-la com o reconhecedor de voz, a implementação está demonstrada no código 8:

Código 8: Iniciação do reconhecimento de voz.

```
private void iniciaDeteccao(Intent serviceIntent) {
      reconhecedorAtivacao = ReconhecimentoDeVozFactory.criaReconhecedor(this);
      reconhecedorAtivacao.startListening(ReconhecimentoDeVozFactory.criaIntentReconhecimentoVoz());
}

public static SpeechRecognizer criaReconhecedor(Context context) {
        ReconhecimentoVozListener ativacaoReconhecimento = new ReconhecimentoVozListener(context);
        SpeechRecognizer speecher = SpeechRecognizer.createSpeechRecognizer(context);
        speecher.setRecognitionListener(ativacaoReconhecimento);
        return speecher;
}
```

No código 8 o reconhecedor de voz iniciou a *Intent* com as configurações descritas e os comandos falados pelo usuário são enviados para a classe *ReconhecimentoVozListener*, que foi configurada dentro do reconhecedor através do método set.

A classe *ReconhecimentoVozListener* precisa implementar a interface *RecognitionListener* que obriga a implementação vários métodos, porém, o método de mais importância para a aplicação é o *onResults*, que é chamado quando o reconhecedor entende que o comando está terminado, ou seja, que o usuário terminou de pronunciar um comando.

Este método traz uma lista de frases ordenadas pela porcentagem de probabilidade da frase entendida, a implementação do método pode ser vista no código 9:

Código 9: Implementação do método onResults da interface RecognitionListener

```
@Override
    public void onResults(Bundle results) {
        if ((results != null) && results.containsKey(SpeechRecognizer.RESULTS_RECOGNITION)) {
            List<String> heard = results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION);
            if (!heard.isEmpty()) {
                processaResultado(heard);
            }
            ativa();
        } else {
            Log.d(TAG, "no results");
        }

    }
```

O método processaResultado() demonstrado no código 9, faz diversas verificações para tentar entender o que o usuário deseja que a aplicação faça e o método ativa() reativa o reconhecimento de voz automaticamente para que o usuário continue trabalhando com a aplicação sem interferir no dispositivo.

5.4 TRANSFORMAÇÃO DA LINGUAGEM NATURAL EM COMANDOS

Desenvolveu-se uma técnica para transformar as frases do usuário em comandos executáveis pela aplicação, esta técnica é composta por quatro passos, enumeração dos comandos que a aplicação pode executar, listagem das palavras chaves que o usuário pode

utilizar, e indexação das palavras semelhantes que possuem mesmo significado, ou seja, que executam o mesmo comando, e por fim, a associação dos comandos com as palavras indexadas.

No primeiro passo desenvolveu-se 8 comandos que podem ser executados pela aplicação:

- Locução das obras disponíveis
- Locução das unidades construtivas da obra
- Pesquisa por obra específica
- Pesquisa por tarefa na obra inteira (sem conhecimento da unidade construtiva)
- Pesquisa por tarefa na unidade construtiva da obra
- Inclusão de medição
- Visualização da medição
- Visualização das informações de medição da tarefa

O segundo passo foi listar as palavras chaves que o usuário pode querer utilizar, sendo elas: obra, obras, unidades construtivas, unidade construtiva, unidade, pesquisar, pesquisa, procurar, encontrar, informações, informação, tarefa, item, serviço, visualizar, listar, apresentar, informar.

O terceiro passo foi indexar essas palavras por objetivo considerando o contexto da aplicação:

Quadro 4: Indexação de palavras por objetivo

Código de indexação	Palavras
1	Obra, obras
2	Unidades construtivas, unidade construtiva, unidade
3	Pesquisar, pesquisa, procurar, encontrar
4	Informações, informação
5	Tarefa, item, serviço
6	Visualizar, listar, apresentar, informar
7	Incluir, criar, adicionar
8	Medição, nova medição, outra medição

Fonte: Elaborada pelo autor

No quarto passo é necessário realizar as combinações de palavras que podem ser comandos e nesse caso, combinações diferentes podem executar o mesmo comando.

A combinação ocorre pela combinação dos códigos de indexação na ordem da frase onde o número 0 representa a ausência de palavras.

Por exemplo: 6100, a combinação 6100 significa a combinação das palavras do código 6 com as palavras do código 1, qualquer combinação entre ambos irá gerar um único comando.

Com isso, tem-se o seguinte quadro:

Quadro 5: Combinações das palavras

Combinação	Comando
6100	Locução das obras disponíveis
6200	Pesquisa por unidade construtiva específica da obra
3510	Pesquisa por tarefa na unidade na obra inteira (sem conhecimento da unidade construtiva)
7800	Inclusão de uma nova medição
6852	Visualização das informações de uma medição
6450	Visualização das informações da tarefa
9100	Seleciona a obra
9200	Seleciona a unidade construtiva

Fonte: Elaborada pelo autor

Existe uma classe enumeradora de comandos, uma pré-indexação de combinações e comandos com chave e valor, onde a chave é a combinação e o valor é um dos comandos no enumerador, como demonstrado no código 10:

Código 10: Comandos indexados por combinação

```
public static Map<String, ComandosSienge> getRede() {
	Map<String, ComandosSienge> comandos = new HashMap<String, ComandosSienge>();
	comandos.put("6100", ComandosSienge.VISUALIZACAO_DE_OBRAS);
	comandos.put("6200", ComandosSienge.VISUALIZACAO_DE_UNIDADES_CONSTRUTIVAS);
	comandos.put("3510", ComandosSienge.PESQUISA_POR_TAREFA_NA_OBRA_INTEIRA);
	comandos.put("7800", ComandosSienge.INCLUSAO_MEDICAO);
	comandos.put("6852", ComandosSienge.VISUALIZACAO_MEDICAO);
	comandos.put("6450",ComandosSienge.VISUALIZACAO_DAS_INFORMACOES_DA_TAREF
A);
	comandos.put("9100", ComandosSienge.SELECIONAR_OBRA);
	comandos.put("9200", ComandosSienge.SELECIONAR_UNIDADE_CONSTRUTIVA);
	return comandos;
}
```

O código 11 demonstra como foi implementada a indexação das palavras por significado.

Código 11: Implementação da indexação das palavras por significado

```
public static Map<Integer, List<String>> getPalavrasPreProgramadas() {
	tipoPalavras = new HashMap<Integer, List<String>>();
	tipoPalavras.put(1, Arrays.asList("obra", "obras"));
	tipoPalavras.put(2, Arrays.asList("unidades construtivas", "unidade construtiva", "unidade",
"unidades"));
	tipoPalavras.put(3, Arrays.asList("pesquisar", "pesquisa", "procurar", "encontrar"));
	tipoPalavras.put(4, Arrays.asList("informações", "informação"));
	tipoPalavras.put(5, Arrays.asList("tarefa", "item", "serviço"));
	tipoPalavras.put(6, Arrays.asList("visualizar", "listar", "apresentar", "informar"));
	tipoPalavras.put(7, Arrays.asList("incluir", "criar", "adicionar"));
	tipoPalavras.put(8, Arrays.asList("medição", "nova medição", "outra medição"));
	tipoPalavras.put(9, Arrays.asList("selecionar", "utilizar", "acessar", "selecione", "acesse"));
	return tipoPalavras;
}
```

O método demonstrado no código 12 consome as preparações do código 11, transformando o comando de voz em combinações que correspondem a comandos ou não.

Código 12: Método principal responsável pela transformação do comando em uma combinação.

```
private String transformaComandoEmCombinacao(ComandoVoz comando) {
        comando.prioriza(getTodasAsPalavrasPreProgramadas());
        String[] palavras = comando.getPrimeiraCorrespondencia().split(" ");
        String combinacao = "";
        for (String palavra : palavras) {
                if(palavraDentreOsTiposEsperados(palavra)) {
                        combinacao = combinacao + getChaveDaPalavra(palavra);
                }
        }

        return combinacao.length() == 4 ? combinacao : preencheCombinacaoComZeros(combinacao);
}
```

Estes são os códigos, que fazem parte do desenvolvimento do protótipo, bem como as combinações mostradas nos quadros que servem para o devido funcionamento da estrutura de reconhecimento de voz.

5.5 PROTÓTIPO

Neste, é desenvolvido um protótipo do produto e está enumerado abaixo as funcionalidades presentes neste protótipo.

5.5.1 Funcionalidade

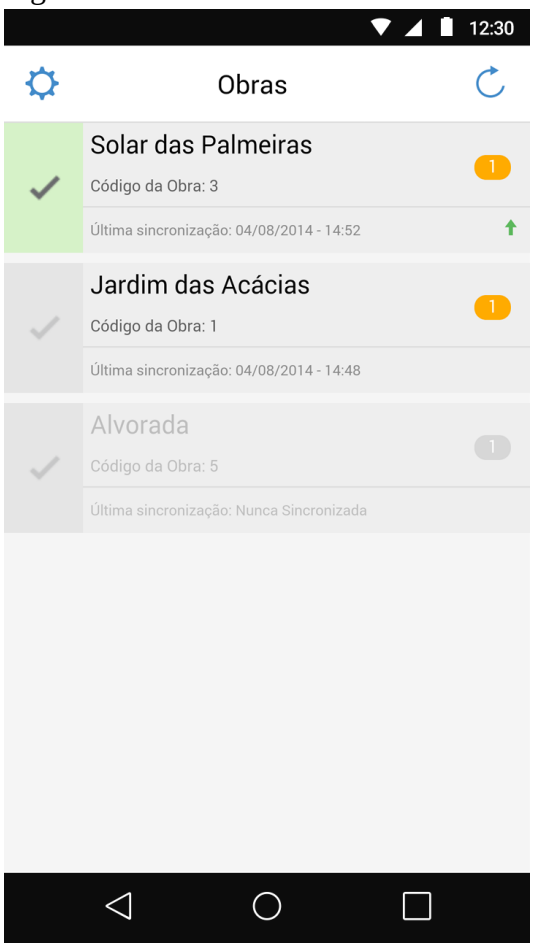
Funcionalidades do protótipo ilustradas com figuras e descritivo das mesmas, relacionando todas as interações que podem ocorrer.

5.5.1.1 Chamativas para operações

O sistema apresenta a tela para escolher a (s) obra (s) alocadas para este usuário, conforme representado na figura 19, porém este é o sistema Sienge Mobile e não a tela do protótipo. O protótipo realiza as chamativas da mesma forma que um usuário utilizaria esta tela, porém com comandos de voz.

O sistema fica aguardando o usuário falar qual obra ele gostaria de abrir para realizar a inclusão de medições. Assim que o sistema entender o que o usuário falou, prepara a abertura da obra. Comando a utilizar: Selecionar Obra “Noma da Obra”.

Figura 19: Tela de escolha da obra



Fonte: Imagem capturada pelo Autor

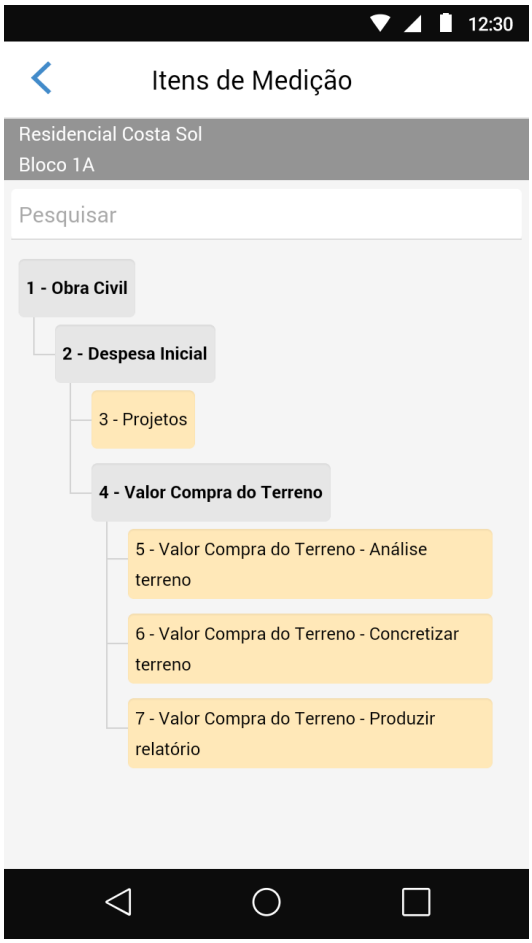
Conforme representado na figura 19, o sistema aguarda até que o usuário fale a obra que deseja abrir.

5.5.1.2 Itens de Medição

Na figura 20 é representado o sistema Sienge Mobile, demonstrando os itens de medição da obra. O protótipo, ao invés de demonstrar estes itens, aguarda a escolha do item a medir, existente na obra escolhida anteriormente.

Neste momento da aplicação o sistema aguarda o usuário falar a unidade construtiva que gostaria de realizar a inclusão de medições daquela obra. Comando a utilizar: Selecionar Unidade Construtiva “Nome unidade construtiva”.

Figura 20: Tela de escolha do item de medição



Fonte: Imagem capturada pelo Autor

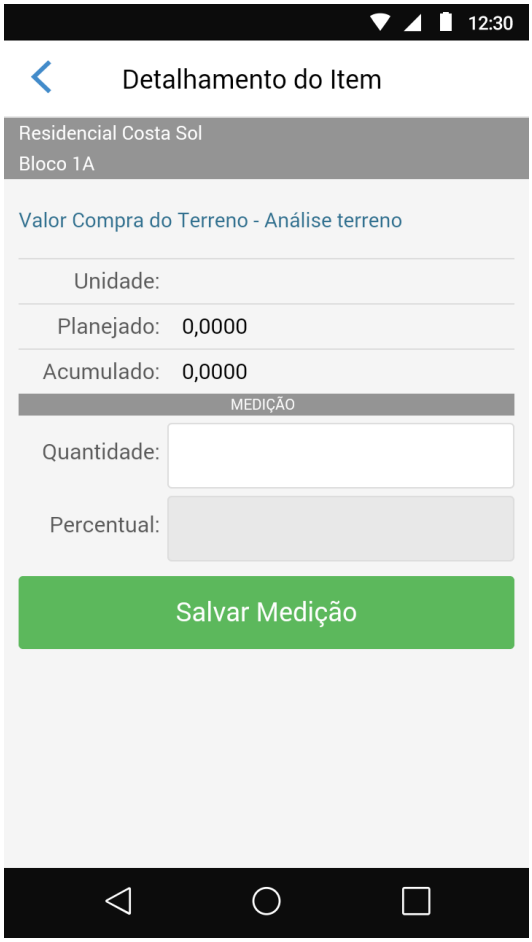
Conforme representado na figura 20, o sistema aguarda até que o usuário fale a unidade construtiva daquela obra escolhida anteriormente, para cadastro de medições.

5.5.1.3 Medição

Na figura 21 é representado o sistema Sienge Mobile, demonstrando o item escolhido. O protótipo, ao invés de mostrar este item, aguarda o comando de voz informando a quantidade da medição daquele item escolhido anteriormente.

Comando a utilizar: Inclusão de Medição. Assim, o sistema inicia a inclusão da medição, verificando, primeiramente, se já existe uma unidade construtiva e uma obra, solicitando, assim, o nome ou código da tarefa, mas caso ainda não tenha obra ou unidade construtiva selecionada, o sistema realiza a solicitação desta que falta. Após informar o nome da tarefa, o programa pergunta se é para medir por quantidade ou por percentual, após a resposta do questionamento, solicita o valor a ser inserido, bem como uma confirmação “Sim” ou “Não” desta operação.

Figura 21: Tela de inclusão de medição



Fonte: Imagem capturada pelo Autor

Conforme representado na figura 21, o sistema aguarda até que o usuário fale a quantidade ou o percentual a ser inserido.

Após este procedimento, o usuário pode visualizar informações das medições já realizadas, utilizando o comando: Visualizar informações da Medição. Assim, o sistema responde as informações da medição de uma tarefa em dois níveis:

- Quantidade Acumulada: Soma das quantidades medidas até o momento;
- Quantidade Planejada: Quantidade que foi planejada para execução daquela tarefa;

Todo o procedimento de inclusão foi realizado, para encerrar o sistema, deve-se utilizar o seguinte comando: Terminar.

5.6 AVALIAÇÃO

Para verificar a eficiência da aplicação, alguns cenários de testes foram produzidos a fim de obter dois indicadores:

- Tempo para se cadastrar um grupo de medições.
- Palavras não interpretadas pelo dispositivo.

Os testes foram realizados com cinco pessoas e o dispositivo foi submetido a locais que possuíam ruídos, como também em locais que não possuíam ruídos. Em ambos os testes, avaliou-se também o dispositivo na forma *online* e *off-line* nestes ambientes.

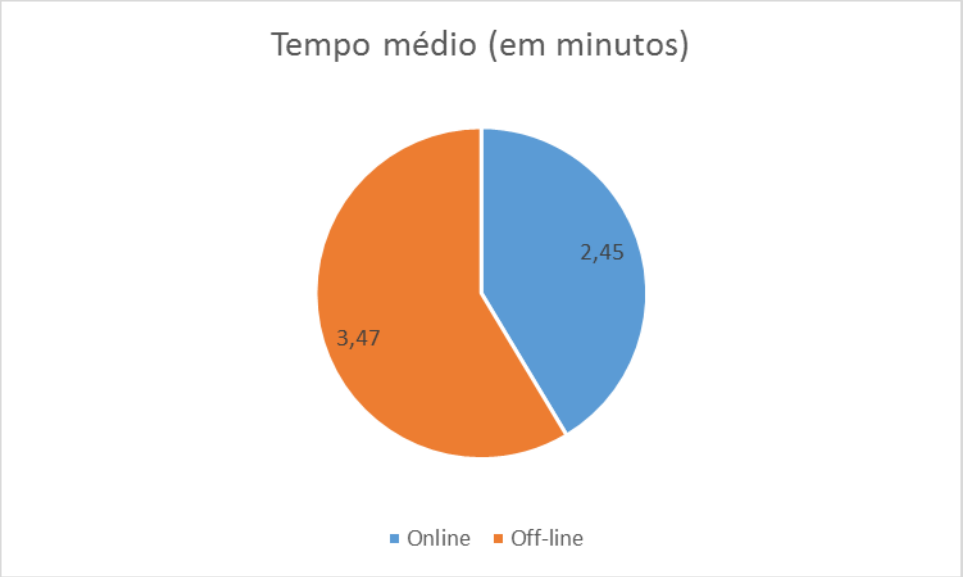
Os cenários de teste visam obter resultados, considerando utilização do software em cenários reais, por este motivo o objetivo não é garantir o funcionamento do software, mas sim analisar como ele se comporta em meio a cenários possivelmente reais, portanto, todos os testes foram realizados utilizando um fone/microfone conectado por *bluetooth*, já que, visando um cenário real, o aplicativo é utilizado para que o usuário não precise usar as mãos, estando provavelmente no bolso ou em algum lugar fora do alcance de sua voz.

Nos testes aplicados, os usuários tinham o objetivo de realizar medições de um conjunto de três tarefas da mesma unidade construtiva da mesma obra.

5.6.1 Testes em um ambiente sem ruído

O protótipo foi submetido a dois testes em um ambiente, onde não apresentava ruído. Nos gráficos 1 e 2 é apresentado o resultado dos testes:

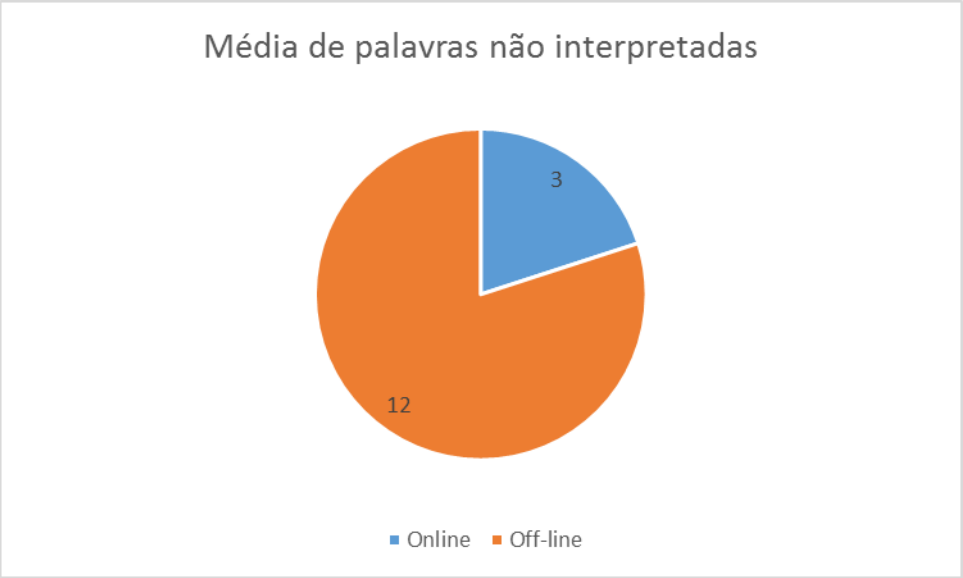
Gráfico 1: Tempo médio de cadastro de medição em ambiente sem ruído



Fonte: Elaborado pelo Autor

Conforme demonstrado no gráfico 1, foi solicitado ao protótipo, realizações de cadastro de grupos de medições, em um ambiente sem ruído. O protótipo teve um tempo médio de 3,47 minutos para realizar esta operação no modo *off-line*, enquanto em modo *online*, teve um tempo médio de 2,45 minutos para realizar esta operação.

Gráfico 2: Palavras não interpretadas em ambiente sem ruído



Fonte: Elaborado pelo Autor

Conforme demonstrado no gráfico 2, foi realizado solicitações ao protótipo em um ambiente sem ruído e o mesmo, teve uma média de 12 palavras não interpretadas em modo *off-line*, enquanto em modo *online*, teve 3 palavras não interpretadas.

5.6.2 Testes em um ambiente com ruído

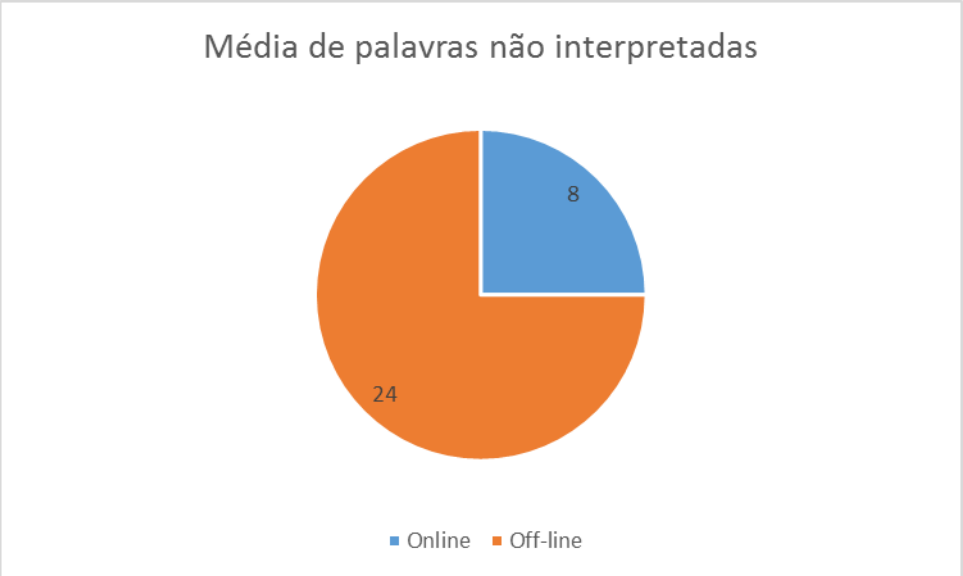
O protótipo foi submetido a dois testes em um ambiente, onde apresentava ruído. Nos gráficos 3 e 4 é apresentado o resultado dos testes:

Gráfico 3: Tempo médio de cadastro de medição em ambiente com ruído

Fonte: Elaborado pelo Autor

Conforme demonstrado no gráfico 3, foi solicitado ao protótipo, realizações de cadastro de grupos de medições, em um ambiente com ruído. O protótipo teve um tempo médio de 5,22 minutos para realizar as operações no modo *off-line*, enquanto em modo *online*, teve um tempo médio de 3,30 minutos para realizar as operações.

Gráfico 4: Palavras não interpretadas em ambiente com ruído



Fonte: Elaborado pelo Autor

Conforme demonstrado no gráfico 4, foi realizado solicitações ao protótipo em um ambiente com ruído e o mesmo, teve uma média de 24 palavras não interpretadas em modo *off-line*, enquanto em modo *online*, teve 8 palavras não interpretadas.

5.6.3 Análise dos testes

Ao analisar os testes, verifica-se que o desempenho do aplicativo quando está *online* é superior quando o mesmo está em modo *off-line* e, isso, se agrava ainda mais quando há ambientes com ruídos, pois a diferença de palavras não interpretadas no teste *off-line* em locais com e sem ruído é de 16 e a diferença de tempo médio é de 2 minutos.

Um questionário foi aplicado aos participantes, com o intuito de saber suas opiniões quanto a utilização do aplicativo em uma obra real. O questionário pode ser visto no quadro abaixo.

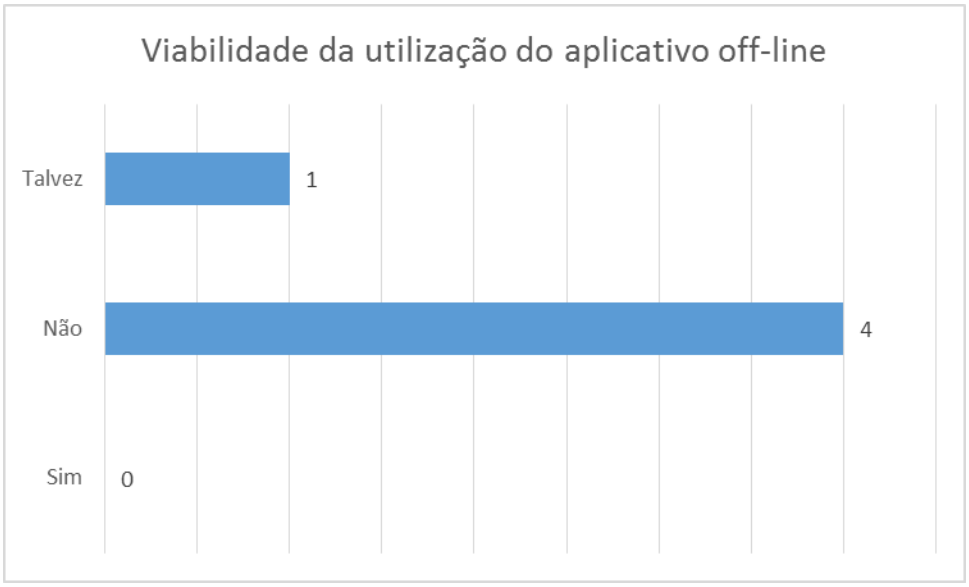
Quadro 6: Perguntas do questionário aplicadas aos participantes

1. Na sua opinião a utilização do aplicativo em uma obra é viável quando está *off-line*?
2. E quando o aplicativo está *online*, como você considera a viabilidade da utilização.
3. Descreva sua experiência como usuário ao utilizar o aplicativo.

Fonte: Elaborado pelo Autor

A partir do questionário disponibilizado aos participantes, as respostas foram importadas em gráficos, conforme pode ser visto abaixo:

Gráfico 5: Viabilidade da utilização do aplicativo off-line



Fonte: Elaborado pelo Autor

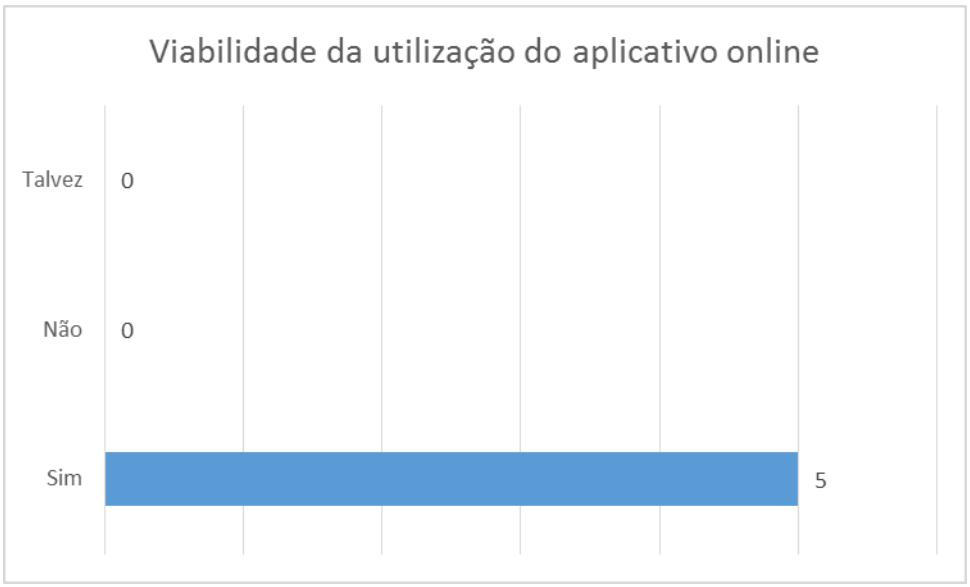
Conforme gráfico 5, os usuários responderam, em sua maioria, que é inviável a utilização do protótipo em modo *off-line*, com a estrutura de desenvolvimento que possui até então. É um resultado esperado, porém ruim.

Um fato interessante sobre a utilização do aplicativo *off-line* foi levantado por um usuário que participou dos testes. Em suas palavras ele comenta:

> "Quando o aplicativo está off-line é muito difícil; fazê-lo entender as obras e tarefas que falamos, para que fosse possível dar prosseguimento na medição eu fui obrigado a utilizar o número da obra e das tarefas ao invés de seus nomes para que o software me entendesse". – Participante #1.

O gráfico 6, apresentado abaixo, representa as respostas dos usuários em relação a utilização do aplicativo *online*.

Gráfico 6: Viabilidade da utilização do aplicativo online



Fonte: Elaborado pelo Autor

No cenário de testes demonstrado no gráfico 6, as respostas foram promissoras, mesmo sendo um protótipo e tendo um vocabulário pequeno de palavras à serem utilizadas, os usuários foram unânimes em dizer que a utilização do aplicativo é viável, porém com algumas ressalvas.

Dois dos usuários participantes, tiveram ressalvas quanto a utilização da aplicação em ambientes com muito ruído. Para o Participante #5:

> "A utilização do aplicativo em ambientes com muito barulho, se torna um pouco mais cansativa, do que em ambientes, onde não há um nível muito alto de ruídos, porém, não é um impedimento para a utilização do aplicativo. Para resolver este problema basta pronunciar os comandos um pouco mais alto do que o normal, não chega a ser um berro, mas as pessoas em volta podem ouvir os comandos que estou pronunciando". – Participante #5.

Um fato a ser considerado é que a qualidade da recepção da voz do usuário, também depende da qualidade do microfone *bluetooth* ou do celular que está a utilizar, uma vez que, este, pode interferir negativamente ou não na filtragem de ruídos.

Em relação a última pergunta do questionário, os usuários descreveram que ficaram bastante frustrados ao usar o aplicativo *off-line*, mesmo utilizando-se de números para buscar as informações contidas no aplicativo, o software não interpretava corretamente em alguns momentos e os comandos existentes para manusear a ferramenta, são realizados através de palavras, logo, o problema persiste.

Ao falar sobre a experiência de utilizar em modo *online*, os participantes gostaram da forma como o sistema retribui às solicitações e dos comandos criados. O participante #4, descreveu:

> "Gostei da estrutura dos comandos, pelo fato de que posso falar com o aplicativo como se estivesse falando com outra pessoa e não dando comandos à um computador, por exemplo, o comando de selecionar obra, no qual posso dizer: 'Selecione a obra X'". – Participante #4.

Já o participante #1 têm ressalvas sobre as retribuições das solicitações, ele explica:

> "As retribuições que o sistema proporciona às minhas solicitações são claras, porém demoram a cada nova medição. Preciso escutar o software me solicitar se quero realizar a medição por quantidade ou por percentual, depois de um tempo isso fica chato de ouvir. Se eu já conheço o programa, eu posso dizer o tipo de medição que quero fazer no início do comando, sem que o software precise me pedir, agilizando, assim, o processo". – Participante #4.

Além desses fatos apresentados, os participantes comentaram também, que devem sempre esperar a aplicação soar o bip para que o próximo comando ou resposta possa ser pronunciado. Alguns dos participantes, que não estavam acostumados com tecnologias de reconhecimento de voz, pronunciaram comandos antes do bip ser soado, o que ocasiona na necessidade de repetir o comando. Descreveram que quando o aplicativo solicita a próxima informação, eles automaticamente respondem ao invés de esperar o bip soar. O participante #4 propôs uma melhoria para esta situação:

> "Percebi que após o programa me solicitar uma resposta, o tempo que ele leva para dar o bip e iniciar novamente o reconhecimento, é um pouco alto. Caso este tempo fosse menor ou imediatamente após o aplicativo terminar de solicitar a próxima informação, eu não teria tido que repetir algumas vezes". – Participante #4.

O participante #3 concorda que isso realmente ocasiona a necessidade de realizar a repetição de algumas respostas, mas tem uma consideração diferente sobre este problema:

> "Durante os testes, assim que o aplicativo me solicitava uma informação, meu primeiro impulso era respondê-lo, mas não funcionava, pois ele ainda não estava me ouvindo. Como não estou acostumado a utilizar este tipo de tecnologia, acredito que seja questão de costume a adaptação de esperar o bip soar para pronunciamento do próximo comando". – Participante #3.

Apesar de todos estes comentários, os participantes ressaltaram da viabilidade do protótipo em uma obra real.

6 CONCLUSÕES E TRABALHOS FUTUROS

Através do referencial teórico é possível concluir que quando implementadas novas tecnologias, por meio de inovação dos processos internos e externos de uma organização, a mesma é beneficiada em questões de reestruturação dos métodos utilizados no dia-a-dia de seus trabalhadores, proporcionando qualidade de desenvolvimento e qualidade de execução.

Tendo em vista o quanto o reconhecimento de voz pode ajudar e melhorar nas atividades de um processo, foi desenvolvida uma aplicação para a plataforma Android, a qual dispõe de uma programação de reconhecimento e entendimento de voz para os processos de uma construção civil, para auxiliar no desempenho e cumprimento dos projetos que a envolvem.

Os objetivos específicos do presente trabalho foram atingidos e mostrados no decorrer de cada capítulo deste trabalho de conclusão de curso. O objetivo de apresentar sistemas e bibliotecas para reconhecimento de voz e analisar os processos relacionados com o dia-a-dia do gestor de mão de obra, foram atingidos através da realização da pesquisa bibliográfica apresentada no capítulo 2. O objetivo de realizar uma modelagem de um protótipo funcional de aplicação móvel foi atingido através da elaboração de diagramas de requisitos, casos de uso, de classe e de deployment presentes no capítulo 4. Por final, os objetivos de desenvolvimento do protótipo e avaliação do mesmo, foram atingidos através da listagem de códigos utilizados e pela sujeição de testes realizados, apresentando gráficos das ocorrências obtidas, bem como as respostas dos questionamentos realizados com os participantes, presentes no capítulo 5.

Como principal contribuição desse trabalho, é possível verificar a viabilidade descrita pelos participantes de o protótipo poder ser utilizado em uma obra real, realizando consultas das medições inseridas em uma obra, bem como inserções de medições apenas por comandos de voz.

Através disto, este protótipo proporciona agilidade no processo de informar o dia-a-dia da obra, aprimorando o registro destas informações e assim, proporciona um diferencial competitivo no mercado em que atua.

Para trabalhos futuros, pretende-se automatizar a seleção da obra através do GPS, identificando que o dispositivo está entrando na zona de uma obra e, assim, selecioná-la automaticamente para que o usuário apenas tenha que informar a unidade construtiva na realização de uma medição. Também pretende-se desvincular o protótipo do sistema Sienge

Mobile, ou seja, executá-lo de forma independente e, assim, ser usado por múltiplas empresas que não possuem o ERP Sienge.

Propõe-se criar uma funcionalidade de pause na atividade de reconhecimento do comando a ser falado pelo usuário, isto, pois, quando um usuário for solicitar uma operação ao protótipo e no mesmo momento ocorre de alguém vir falar com este usuário ou uma outra atividade qualquer venha a ser elencada para execução por este usuário, o mesmo precisa pausar o reconhecimento e tratar tais situações momentâneas. Quando finalizadas, pode acionar a funcionalidade de reconhecimento, retornando sua solicitação ao protótipo. Propõe-se também, a realização de testes com mais participantes e com mais variedades de versões do sistema operacional Android, pois no presente trabalho, por questões de indisponibilidade de outros participantes, bem como a não obtenção de dispositivos com versões diferentes, não se pôde realizar tais testes.

Outros dois pontos importantes é ampliar esta funcionalidade para IOS, o qual está desenvolvida apenas para Android e analisar para uma futura implementação, os comentários e *feedbacks* realizados pelos participantes nos cenários de testes.

REFERÊNCIAS

ABLESON, W. Frank et al. Android em ação. **Elsevier**, Rio de Janeiro, 2012. Disponível em: <https://books.google.com.br/books?id=zQLJY6KCFo8C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 26 out. 2015.

ANDRADE, Antonio Luis Lordelo. Usabilidade de Interface Web – Avaliação heurística no jornalismo on-line. **E-papers Serviços Editoriais Ltda**, Rio de Janeiro, 2007. Disponível em: <https://books.google.com.br/books?id=wiF1jPaV8_IC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 03 mai. 2015.

A SOFTPLAN: **Quem somos**. Disponível em: <http://www.softplan.com.br/a-softplan/quem-somos/>. Acessado em: 02 nov. 2014.

ASSIS, Maria Cristina de. **Metodologia do trabalho científico**. Disponível em: <http://portal.virtual.ufpb.br/biblioteca-virtual/files/pub_1291081139.pdf>. Acesso em: 26 out. 2014.

BALTZAN, Paige et al. Sistemas de Informação. **McGraw Hill Brasil**, Nova Iorque, 2012. Disponível em: <https://books.google.com.br/books?id=NJkR83DSkPYC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 20 mai. 2015.

BARBOSA, Simone Diniz Junqueira et al. Interação Humano-Computador. **Elsevier Editora Ltda**, São Paulo, 2010. Disponível em: <https://books.google.com.br/books?id=qk0skwr_cewC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 03 mai. 2015.

BERNSTEIN, Harvey M. e LAQUIDARA-CARR, Donna. Information mobility: Improving Team Collaboration Through The Movement of Project Information. **Bedford**, Massachusetts, 2013. Disponível em: <http://bradleybim.files.wordpress.com/2013/12/2013_information_mobility_bim_smart_market_report.pdf>. Acesso em: 26 out. 2014.

BRASIL. IBGE. (Org.). **Pesquisa anual da indústria da construção.** 2012. Disponível em: <http://www.cbicdados.com.br/media/anexos/PAIC2012.pdf>. Acesso em: 02 nov. 2014.

BOOCH, Grady et al. UML: Guia do Usuário. **Elsevier**, Rio de Janeiro, 2005. Disponível em: < https://books.google.com.br/books?id=ddWqxcDKGF8C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 20 mai. 2015.

CASSIMIRO, Flávio Renato. **Benefícios em Implantar Sistema de Gestão de Qualidade em Empresas da Construção Civil.** 2013. Disponível em: <http://www.techoje.com.br/site/techoje/categoria/detalhe_artigo/1660>. Acessado em: 14 set. 2014.

CINAR, Onur. Pro Android C++ with the NDK. **Apress**, Nova Iorque, 2012. Disponível em: < https://books.google.com.br/books?id=BaZYxBdLKH8C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 24 mai. 2015.

COSTA, Daniel Gouveia. Java em rede: recursos avançados de programação. **Brasport**, Rio de Janeiro, 2008. Disponível em: <https://books.google.com.br/books?id=W4oBsv7lifMC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 25 out. 2015.

DEITEL, Paul et al. **Java: Como Programar**. Pearson Prentice Hall, São Paulo, 2010. ISBN 978-85-7605-194-7

DEITEL, Paul et al. Android para programadores: Uma abordagem baseada em aplicativos. **Bookman**, Porto Alegre, 2012. Disponível em: < https://books.google.com.br/books?id=-wz1BgAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 24 mai. 2015.

FIELDING, T. **Architectural Styles and the Design of Network-based Software Architectures.** 200. 180p Tese (Doutorado). University of Califórnia, Irvine, 2000.

FILHO, Luiz Carlos Querino. Desenvolvendo seu primeiro aplicativo Android. **Novatec Editora Ltda,** São Paulo, 2014. Disponível em: <https://books.google.com.br/books?id=XUSZAwAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 26 out. 2015.

GAMMA, Erich et al. Padrões de Projeto: Soluções reutilizáveis de software orientado a objetos. **Bookman**, Porto Alegre, 2009. Disponível em: <https://books.google.com.br/books?id=U91CYCqTCgkC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 26 out. 2015.

HILL, Simon. 2010. **History of Android: First Applications Prototypes & Other Events** Disponível em < http://www.brighthub.com/mobile/google-android/articles/18260.aspx >. Acessado em: 02 nov. 2014.

HUANG, X.; Acero, A.; Hon, H. 2001. **Spoken Language Processing. A guide to theory, algorithm and system development**. Prentice Hall PTR. ISBN 0-13-022616-5.

JUNIOR, Cícero Caiçara. Sistemas integrados de gestão ERP: Uma abordagem gerencial. **Editora Ibpex**, Curitiba, 2008. Disponível em: <https://books.google.com.br/books?id=Fy9dO9Wx_D8C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 20 mai. 2015.

KERIEVISKY, Joshua. Refatoração para padrões. Porto Alegre: **Bookman** 2008. Disponível em: < https://books.google.com.br/books?id=C4u36j_WSuIC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 26 out. 2015.

LALANNE, Denis et al. Human Machine – Research Results of the MMI Program. **Springer**, Berlin, 2009. Disponível em: <https://books.google.com.br/books?id=sbgmDm9W6GkC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 03 mai. 2015.

LECHETA, R. Ricardo. Google Android: Aprenda a criar aplicações para dispositivos móveis com o Android SDK. **Novatec**, São Paulo, 2013. Disponível em: < https://books.google.com.br/books?id=NrVUAwAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 24 mai. 2015.

LENZI, Fernando César et al. Ação empreendora: como desenvolver e administrar o seu negócio com excelência. **Editora Gente**, São Paulo, 2010. Disponível em: <https://books.google.com.br/books?id=9x9PR9l6ntUC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 26 out. 2015.

LIMA, Isaías et al. Inteligência Artificial. **Elsevier Editora Ltda**, Rio e Janeiro, 2014. Disponível em: < https://books.google.com.br/books?id=qjJeBgAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 17 mai. 2015.

LOBO, Edson J. R. Curso de Engenharia de Software – Métodos e processos para garantir a qualidade no desenvolvimento de software. **Digerati Books**, São Paulo, 2008. Disponível em: < https://books.google.com.br/books?id=ZJznA9UrtVAC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 14 out. 2015.

MARTINS, José Carlos Cordeiro. Técnicas para Gerenciamento de Projetos de Software. **Brasport Livros e Multimídia Ltda**, Rio de Janeiro, 2007. Disponível em: < https://books.google.com.br/books?id=Axl2RZQdE68C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 14 out. 2015.

MATTOS, Érico Casella Tavares de. Programação de softwares em Java. **Digerati Books**, São Paulo, 2007. Disponível em: <https://books.google.com.br/books?id=urix0DCtbIcC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado: 26 out. 2015.

McGraw Hill Construction. **Information Mobility: Improving Team Collaboration Throught the Movement of Project Information**, 2013 Disponível em: <http://enr.construction.com/engineering/pdf/News/Information_Mobility_SMR_2013.pdf >. Acessado em: 26 out. 2014.

MILETTO, Evandro Manara e outros. Desenvolvimento de Software II: Introdução ao Desenvolvimento Web com HTML, CSS, JavaScript e PHP - Eixo: Informação e Comunicação – Série Tekne. **Bookman**, Porto Alegre, 2014. Disponível em: <https://books.google.com.br/books?id=lcLFAwAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 25 out. 2015

O SIENGE: Suas operações sob controle onde você estiver. Disponível em: <http://www.sienge.com.br/o-sienge/>. Acessado em: 02 nov. 2014.

PAPAPETROU, Patroklos et al. Android Application Development with Maven. **Packt Publishing Ltd**, Livery Place, 2015. Disponível em: < https://books.google.com.br/books?id=DZ96BwAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 24 mai. 2015.

PERDIGÃO, Dulce Montella et al. Teoria e prática da pesquisa aplicada. **Elsevier**, Rio de Janeiro, 2012. Disponível em: <https://books.google.com.br/books?id=Oot9S0RLWjwC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 26 out. 2015.

PEREIRA, André Maués Brabo. **Modelagem de Software Orientada a Objetos com UML**. Disponível em: < http://webserver2.tecgraf.puc-rio.br/ftp_pub/lfm/CIV2802-131-Aula04-ModelagemOrientadaObjetos.pdf >. Acessado em: 27 nov. 2014.

PEREIRA, Lúcio Camilo Oliva et al. Android para Desenvolvedores. **Brasport Livros e Multimídia Ltda**, Rio de Janeiro, 2009. Disponível em: < https://books.google.com.br/books?id=8u9wJowXfdUC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 14 out. 2015.

PEURIFOY, Robert L. et al. Planejamento, equipamentos e métodos para a construção civil. **Mc Graw Hill Brasil**, Porto Alegre, 2015. Disponível em: <https://books.google.com.br/books?id=wyJmCgAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 23 out. 2015.

PIO, Bruno Luiz de Assis. **Uma Introdução às Redes Neurais Artificiais para Estudos em Ecologia**. Disponível em: <https://books.google.com.br/books?id=mr0WOJiu1WEC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 17 mai. 2015.

PONTES, Roberto. Inteligência Artificial nos Investimentos. **Clube de Autores**, Rio de Janeiro, 2011. Disponível em: < https://books.google.com.br/books?id=cJJMBQAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 17 mai. 2015.

PRADO, Sérgio. **Introdução ao funcionamento interno do Android**. Disponível em: < http://sergioprado.org/introducao-ao-funcionamento-interno-do-android/ >. Acessado em: 02 nov. 2014.

PREECE, Rogers Sharp. Design de Interação – Além da interação humano-computador. **Bookman Editora LTDA**, São Paulo, 2011. Disponível em: < https://books.google.com.br/books?id=d_s4AgAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 03 mai. 2015.

PRESSMAN, S. Roger. Engenharia de software: Uma abordagem profissional. **McGraw Hill Brasil**, São Paulo, 2011. Disponível em: < https://books.google.com.br/books?id=y0rH9wuXe68C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 24 mai. 2015.

RAMOS, Ricardo Argenton. Treinamento Prático em UML – Desenvolva e Gerencie Seus Projetos Com Essa Sensacional Ferramenta. **Digerati Books**, São Paulo, 2006. Disponível em: < https://books.google.com.br/books?id=cE4qBWwJM1sC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 20 mai. 2015.

REZENDE, Denis Alcides. Engenharia de Software e Sistemas de Informação. **Brasport**, Rio de Janeiro, 2005. Disponível em: <https://books.google.com.br/books?id=rtBvl_L-1mcC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 23 out. 2015.

ROSS, Julio. Alarmes – Conheça os dispositivos eletrônicos de alarmes para: incêndio, residência, comercial, automotivo e pessoal. **Antenna Edições Técnicas**, Rio de Janeiro, 2008. Disponível em: < https://books.google.com.br/books?id=XcI1BI45rK4C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 03 mai. 2015.

SAMPAIO, Cleuton. Guia do Java: Enterprise Edition 5: desenvolvendo aplicações corporativas. **Brasport**, Rio de Janeiro, 2007. Disponível em: < https://books.google.com.br/books?id=uC0ly5no0s0C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 20 mai. 2015.

SANTOS, Emilio Moreira. Engenharia Linguística – Tecnologias para Apoiar as Decisões Gerenciais na Era da Internet. **E-papers Serviços Editoriais Ltda**, Rio de Janeiro, 2008. Disponível em: < https://books.google.com.br/books?id=q4qnsf4c9qcC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false >. Acessado em: 03 mai. 2015.

SCHMITT, Alexander et al. Towards Adaptive Spoken Dialog Systems. **Springer**, Nova Iorque, 2013. Disponível em: <https://books.google.com.br/books?id=dEACYgnAq28C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 03 mai. 2015.

SERSON, Roberto Rubinstein. Programação orientada a objetos com Java. **Brasport**, Rio de Janeiro, 2007. Disponível em: <https://books.google.com.br/books?id=CsGtipt1wsQC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 20 mai. 2015.

SIENGE: Acompanhamento. Disponível em: <http://www.sienge.com.br/engenharia/#acompanhamento>. Acessado em: 02 nov. 2014.

SIENGE MOBILE: Registro de medição física. Disponível em: <http://www.sienge.com.br/aplicativos/>. Acessado em: 02 nov. 2014.

SILVA, Antonio Carlos Teixeira Da. Inovação: Como criar ideias que geram resultados. **Qualitymark**, Rio de Janeiro, 2003. Disponível em: <https://books.google.com.br/books?id=QlZZsNvnzEYC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 25 out. 2015.

SOMERA, Guilherme. Treinamento Profissional em Java. **Digerati Books**, São Paulo, 2006. Disponível em: <https://books.google.com.br/books?id=QjfF0RKIRO0C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 20 mai. 2015.

STRICKLAND, Jonathan. **Como funciona o Android (Google Phone)**. Disponível em: < http://tecnologia.hsw.uol.com.br/google-phone.htm >. Acessado em: 02 nov. 2014.
TAYLOR, Paul. 2009. ***Text-to-Speech Synthesis*.** Cambridge. Cambridge University Press. ISBN-13: 9780521899277.

TITTEL, ed et al. Html, Xhtml & Css Para Leigos. **Alta Books**, Rio de Janeiro, 2014. Disponível em: <https://books.google.com.br/books?id=BUmXBAAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 20 mai. 2015.

TRÍAS DE BES, Fernando et al. A Bíblia da Inovação. **Leya**, São Paulo, 2012. Disponível em: < https://books.google.com.br/books?id=X9xVkp5TWUQC&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 25 out. 2015.

TURBAN, Efraim et al. Tecnologia da Informação para Gestão – Transformando os Negócios na Economia Digital. **Artmed Editora AS**, São Paulo, 008. Disponível em: <https://books.google.com.br/books?id=HB_PI4-GnDoC&dq=voz+e+processamento+de+fala&hl=pt-BR&source=gbs_navlinks_s>. Acessado em: 03 mai. 2015.

XAVIER, Carlos Magno da Silva et al. Gerenciamento de Projetos de Construção Civil: uma adaptação da metodologia Basic Methodware. **Brasport**, Rio de Janeiro, 2014. Disponível em: <https://books.google.com.br/books?id=1ZWKAwAAQBAJ&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acessado em: 23 out. 2015.