UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

LAURA WOBETO

SISTEMA DE NOTIFICAÇÃO DINÂMICO PARA O TRANSPORTE PÚBLICO DE CURITIBA - SiNDi

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2015

LAURA WOBETO

SISTEMA DE NOTIFICAÇÃO DINÂMICO PARA O TRANSPORTE PÚBLICO DE CURITIBA - SiNDi

Trabalho de Conclusão de Curso, apresentado à disciplina de Trabalho de Conclusão de Curso 2, do curso de Engenharia de Computação dos Departamentos Acadêmicos de Informática – DAINF – e Eletrônica – DAELN – da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Engenheira de Computação.

Orientador: Prof. Dr. Marcelo Rosa.
Co-Orientador: Prof. Dr. Ricardo Lüders.

CURITIBA
2015

Dedico este trabalho à Deus, que apesar da sua Onipotência, nos permite ser engenheiros.

AGRADECIMENTOS

A Deus por ter me dado saúde e força para superar as dificuldades.

Aos professores Marcelo Rosa e Ricardo Lüders, pela orientação, apoio, paciência e confiança.

A todos os demais professores por me proporcionarem o conhecimento racional, a manifestação do caráter e afetividade no processo da formação profissional. Não existem palavras capazes de adjetivar a tamanha excelência dessa equipe docente que terá meu eterno agradecimento.

A URBS, por disponibilizar aos alunos a oportunidade de desenvolver projetos inovadores para melhoria dos serviços prestados a comunidade.

Aos meus pais, Gilberto e Marilú Wobeto, pelo amor, incentivo e apoio incondicional.

Ao time de Retail Customer Service, em especial ao Fabricio Tanko, José Menegotto e Pilipda Samphant, que compreenderam a importância deste trabalho ao permitirem me dedicar a ele em diversos momentos.

Aos familiares, amigos, ao meu namorado e a todos que direta ou indiretamente estiveram presentes nesse processo de formação. Meu muito obrigado.

RESUMO

WOBETO, Laura. Sistema de Notificação Dinâmico para o transporte público de Curitiba - SINDI. 2015. 72 f. Trabalho de Conclusão de Curso (Engenharia de Computação), Universidade Tecnológica Federal do Paraná. Curitiba, 2015.

O sistema de transporte público de Curitiba, para transmitir notificações temporárias aos usuários, utiliza, atualmente, mensagens impressas em papel, expostas em terminais, estações-tubo, pontos de espera e interior dos ônibus. Nesse contexto, o SiNDi (Sistema de Notificação Dinâmico), apresenta um sistema de mensagens baseado na síntese de fala, em que as notificações serão transmitidas por áudio. Desta forma, a comunicação entre a URBS (Urbanização de Curitiba SA), empresa que controla o sistema de transporte público na cidade e a população que utiliza o mesmo, torna-se mais eficiente, por se tratar de uma ação involuntária aos usuários. O projeto de desenvolvimento desse sistema segue as diretrizes do Modelo V de Gestão de Projetos, sendo dividido em três grupos principais: i/ Central; ii/ Sistema Embarcado e iii/ Sistema de Comunicação. A Central é composta pelo computador no qual a URBS expressa a intenção de emitir mensagens para a população. O Sistema Embarcado é composto por um sensor GPS que identifica os locais onde as mensagens deverão ser reproduzidas e uma placa Raspberry PI, Modelo B, que executa a síntese de fala da mensagem enviada pela URBS, reproduzindo-a através de alto-falantes no interior dos ônibus. O Sistema de Comunicação é sem fio que conecta a Central ao Sistema Embarcado, transmitindo textos e proporcionando mobilidade ao sistema. Como resultado, tem-se uma maior flexibilidade na produção e publicação de mensagens na rede de transporte público de Curitiba, aumentando o número de usuários atingidos pelas notificações emitidas pela URBS, assim como aumento na satisfação com o serviço de transporte público.

Palavras-chave: Sistema de Notificação; Transporte Público de Curitiba; Síntese de Fala; Raspberry PI; Comunicação sem fio;

ABSTRACT

WOBETO, Laura. Dynamic Reporting System to the public transport of Curitiba - SINDI. 2015. 72 f. Trabalho de Conclusão de Curso (Engenharia de Computação), Universidade Tecnológica Federal do Paraná. Curitiba, 2015.

The public transport system of Curitiba, to transmit temporary notifications to the users, uses currently messages printed on paper, exposed in terminals, stations-tube, stops and inside the bus. In this context, SiNDi (Dynamic Reporting System) features a messaging system based on speech synthesis, in which the notifications will be transmitted by audio. Thus, communication between URBS (Curitiba Urbanization SA), a company that controls the public transport system in the city and the users become more efficient, because listening is an involuntary action. The software development of this system follows the guidelines of V-Model, which will be divided into three main groups: i/ Center of Operation; ii/ Embedded System and iii/ Communication System. The Center of Operation consists of a computer which URBS express the intention of delivery messages to the population. The Embedded System consists of a GPS sensor which identifies the locations where messages should be played and a Raspberry PI board, Model B, which executes the synthesis of the message sent by URBS, playing it through speakers inside the bus. The Communication System is wireless and aim to connect the Center of Operation to the Embedded System, transmitting text and providing mobility to the system. Overall, the system will provide greater flexibility in the production and publications of messages on public transport network of Curitiba, increasing the number of users affected by the notification issued by URBS, as well as increase the satisfaction with public transport service.

Keywords: Reporting System; Curitiba Public Transport; Speech Synthesis; Raspberry PI; Wireless;

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIAÇÕES, SIGLAS E ACRÔNIMOS

BD	Banco de Dados
CRM	Customer Relationship Management
DAELN	Departamento Acadêmico de Eletrônica
DAINF	Departamento Acadêmico de Informática
GPS	Global Positioning System
IDE	Integrated Development Environment
MBROLA	Multi Band Resynthesis Overlap Add
NTU	Associação Nacional de Transportes Urbanos
PSOLA	Pitch Synchronous Overlap and Add
RIT	Rede Integrada de Transporte
SDK	Software Development Kit
SE	Sistema Embarcado
SiNDi	Sistema de Notificação Dinâmico
TD-PSOLA	Time Domain - Pitch Synchronous Overlap and Add
TTS	Text-To-Speech
URBS	Urbanização de Curitiba SA
UTFPR	Universidade Tecnológica Federal do Paraná

SUMÁRIO

1 INTRODUÇÃO

1.1 Contexto do Tema

Atualmente, é cada vez mais comum encontrar empresas que trabalham meios de melhorar seus relacionamentos com os clientes. O aumento da adesão de sistemas CRM (*Customer Relationship Management*) pode comprovar isso. (BRAMBILLA; PEREIRA; PEREIRA, 2010).

Enquanto 80% dos representantes de grandes empresas acreditam fornecer uma boa experiência ao cliente, apenas 43% dos clientes tiveram efetivamente uma experiência positiva. (ORACLE, 2014).

Em Curitiba, a URBS (Urbanização de Curitiba S/A), sociedade de economia mista com colaboração entre o Estado e empresas particulares (PREFEITURA MUNICIPAL DE CURITIBA, 2014), atende mais de 2,2 milhões de pessoas por dia. (URBS, 2014).

No Brasil, em pesquisa realizada pelo IPEA (Instituto de Pesquisa Econômica Aplicada) em 2011, 31,3% da população afirma que a qualidade do serviço de transporte público em sua cidade é “Regular”.

A NTU (Associação Nacional de Transportes Urbanos) informou que nos últimos dez anos houve uma redução de 30% no número de usuários do transporte público dos principais centros urbanos do país. Em Curitiba foi registrado, entre 2008 e 2011, uma redução de 14 milhões de usuários pagantes transportados (Figura 1) – de 323,50 milhões passou para 309,50 milhões de passageiros por ano, correspondendo a uma queda de cerca de 4%. (ANTONELLI; BARÃO, 2012).

Figura 1 – Número de passageiros do transporte público.
Fonte: Antonelli; Barão, 2012.

Apesar de seu caráter essencial para a população e as cidades, para que não haja uma redução ainda maior, é necessário que o governo invista no setor para que os cidadãos voltem a se interessar na utilização do transporte público. (ANTONELLI; BARÃO, 2012), (DUARTE; LIBARDI; SÁNCHEZ, 2007).

1.2 Caracterização da Oportunidade

As palavras da autora Cristine de Oliveira Lanzoni (2013) apontam explicitamente a oportunidade a ser explorada neste trabalho ao afirmar que:

> O desenvolvimento de sistemas de informação ao usuário configura-se como umas das possíveis estratégias a serem executadas pelo serviço de transporte público, a fim de reverter a sua imagem e, assim, manter e cativar usuários. Um sistema de informação ao usuário tem como objetivo auxiliar todos os usuários a navegarem pelo sistema de transporte público com facilidade. (EMBARQ BRASIL, 2011 apud LANZONI, 2013).

Corrobora com a ideia acima, a afirmação da Associação Internacional de Transportes Públicos de que se faz necessário prover os usuários com informações “essenciais para que os transportes públicos se tornem mais acessíveis e atrativos”. (INTERNACIONAL ASSOCIATION OF PUBLIC TRANSPORT, 2001a, 2001b, 2002). Nesse sentido, um bom material de informação torna o cotidiano das pessoas, que necessitam de informações específicas, mais fácil, ademais

proporcionam aos emissores da informação um bom retorno econômico e de boa credibilidade. (PETTERSSON, 2007).

São exemplos de materiais de informação: diagramas e mapas, tabelas de horários, bilhete, letreiros das estações, quiosques de informação, sinalização, informação em tempo real, avisos sonoros, entre outros. (EMBARQ BRASIL, 2011).

Vivemos em uma era onde novas tecnologias surgem a cada instante, tendo como principal objetivo auxiliar os seres humanos a reduzir custos diversos, seja no trabalho ou na vida pessoal. Contudo, ainda é possível encontrar sistemas e processos arcaicos que não foram modernizados.

Um bom exemplo é o processo de atualização de áudio-mensagens nos ônibus que operam no transporte público de Curitiba, no qual consiste de 4 etapas: i/ definição da mensagem a ser transmitida; ii/ gravação em estúdio da mensagem previamente definida; iii/ gravação de áudio em memória; iv/ alteração da memória nos ônibus.

Ao analisá-lo tecnologicamente, o processo pode ser melhorado significativamente, considerando os seguintes aspectos:

a) Gravação em estúdio: para áudios que o objetivo é informativo e não exibir a qualidade da voz do locutor, métodos mais simplistas podem ser utilizados para substituição dessa etapa. Por exemplo: síntese de fala, gravação em ambiente não acústico, etc.;
b) Gravação de áudio em memória: o áudio pode ser produzido na forma digital, e ser transmitido por qualquer meio de transmissão digital que utilize um protocolo de comunicação: internet, etc.. Ao gravá-lo na memória, a transmissão do áudio passa a exigir movimentação do dispositivo físico e sua execução limitada ao local onde o mesmo se encontra.
c) Alteração de dispositivo: essa etapa deve ser realizada decorrente da etapa anterior. Porém, se o áudio puder ser mantido na forma digital, não é necessária a alteração do dispositivo no local de uso do áudio, e as mensagem podem ser transmitidas via meios de transmissões digitais.

Viu-se então, a oportunidade da reformulação do processo de atualização de áudio-mensagens para a rede de transporte público de Curitiba, através do uso de novas tecnologias existentes no mercado.

1.3 Objetivos

1.3.1 Objetivo Geral

Desenvolver a prova conceitual de um sistema no qual a URBS, de maneira dinâmica, poderá atualizar as áudio-mensagens nos ônibus e difundi-las a um maior contingente de usuários da rede de transporte público de Curitiba, com baixo custo.

O sistema apelidado de **SiNDi**, resume em poucas palavras o objetivo do projeto – **Si**stema de **N**otificação **Di**nâmico.

1.3.2 Objetivos Específicos

1) Desenvolver um sistema flexível, onde a URBS possa alterar, integralmente ou parcialmente, as áudio-mensagens disponíveis nos ônibus;
2) Concepção de um sistema, de acesso remoto, onde não seja necessária a presença física de uma funcionário da URBS no ônibus para alteração das áudio-mensagens;
3) Desenvolver um protocolo de comunicação entre uma central localizada na URBS e o sistema de reprodução de áudio-mensagem;
4) Utilizar coordenadas de GPS para identificar os locais onde as mensagens devem ser reproduzidas;
5) Utilizar um software de síntese de fala no ônibus para reprodução das mensagens enviadas via protocolo de comunicação;
6) Desenvolver o projeto seguindo uma metodologia de desenvolvimento bem definida pelos conceitos de engenharia de software.

1.4 Justificativa

O SiNDi, através de suas características, reduzirá o processo de atualização de áudio-mensagens nos ônibus de quatro ações humanas para uma, trazendo consigo diversos benefícios implícitos sendo estes:

- Independência da URBS com relação ao locutor para gravação de mensagem, tornando o sistema muito mais flexível;
- Inibição da utilização de materiais impressos, poupando assim o meio

ambiente;

- Implementação de comunicação rápida e eficiente, implicando em mensagens atuais e contextualizadas;
- A possibilidade de utilização do sistema de notificações como ferramenta de orientação frente a imprevistos e situações de emergência de maneira mais imediata.

Desta feita, a URBS criará uma mensagem escrita através de um software disponível na Central, que será enviada para o SE (Sistema Embarcado). Ao receber a mensagem, o SE irá armazenar e realizará a síntese de fala quando sua condição para execução for atendida. Nesse momento o sistema a torna uma mensagem de áudio e a reproduz pelos alto-falantes.

De modo geral, a alteração deste processo proporciona redução significativa dos custos, tornando viável o estudo e a implementação do projeto.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Fala Humana

O aparelho fonador, responsável pela fala humana, é composto por um conjunto de órgãos (CUERVO, 2010):

- Pulmões, brônquios, traqueia: produtores da coluna de ar, matéria prima para a produção do som;
- Cordas vocais que se localizam no interior da laringe, produtoras da energia sonora utilizada na fala;
- Cavidades supralaríngeas (faringe, boca e fossas nasais), que funcionam como uma caixa de ressonância: amplificadores do som;
- Lábios, língua, mandíbula: articuladores e transformadores dos sons.

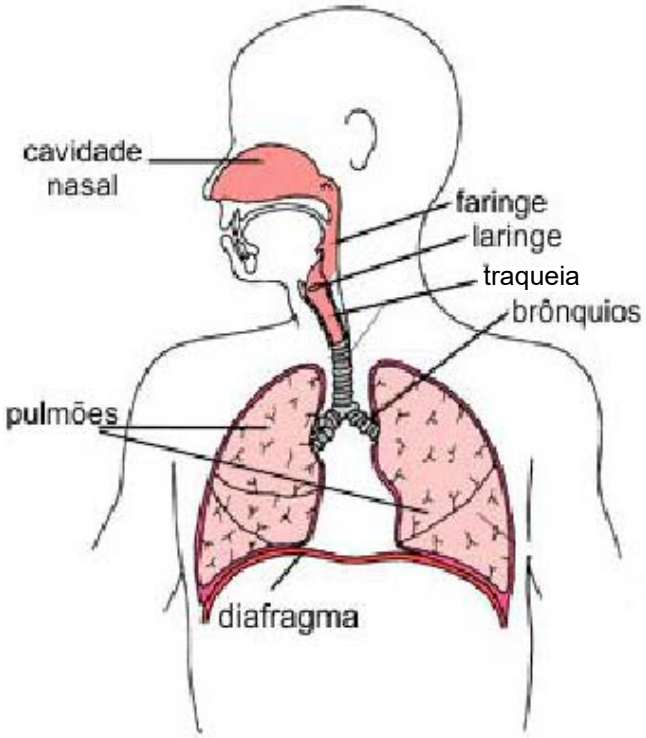

Figura 2 – Órgãos do Aparelho Fonador
Fonte: CUERVO, 2010.

É importante ressaltar que os órgãos do aparelho fonador possuem outras funções, que não a fonação, para ser humano.

Durante a fonação, o som é produzido pela vibração das cordas vocais ao passar o ar expelido pela expiração. A frequência das vibrações se altera conforme as cordas vocais se estiram (Figura 3 - Fonação), ou se abrem. A proporção do estiramento das cordas define a extensão vocal de cada pessoa; vozes agudas tem

menor comprimento que vozes graves. A abertura gerada pelo distanciamento das pregas é chamado de glote. Para respiração e repouso, a glote permanece aberta, obtendo o maior orifício possível entre as pregas vocais. (LIMA, 2015) (ALBUQUERQUE, 2001).

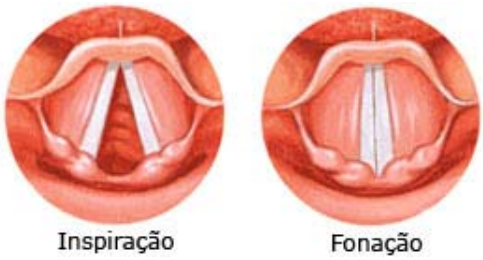

Figura 3 – Cordas Vocais
Fonte: LIMA, s.d.

Após a passagem do ar pela glote, o som é gerado, e antes que possa ser ouvido, sofre um processo de filtragem acústica ao passar pelo trato vocal - órgãos entre a glote e a boca. As características desse processo são determinadas basicamente (NAGLE; CHIQUITO, 1993):

a) pelas posições dos órgãos articuladores;
b) pelas perdas devido à viscosidade das paredes do trato vocal;
c) pelos coeficientes de absorção e reflexão destas paredes;
d) pela existência ou não de acoplamento entre todas as cavidades supralaríngeas.

A voz e o instrumento musical podem ser comparados de acordo com a ilustração a seguir.

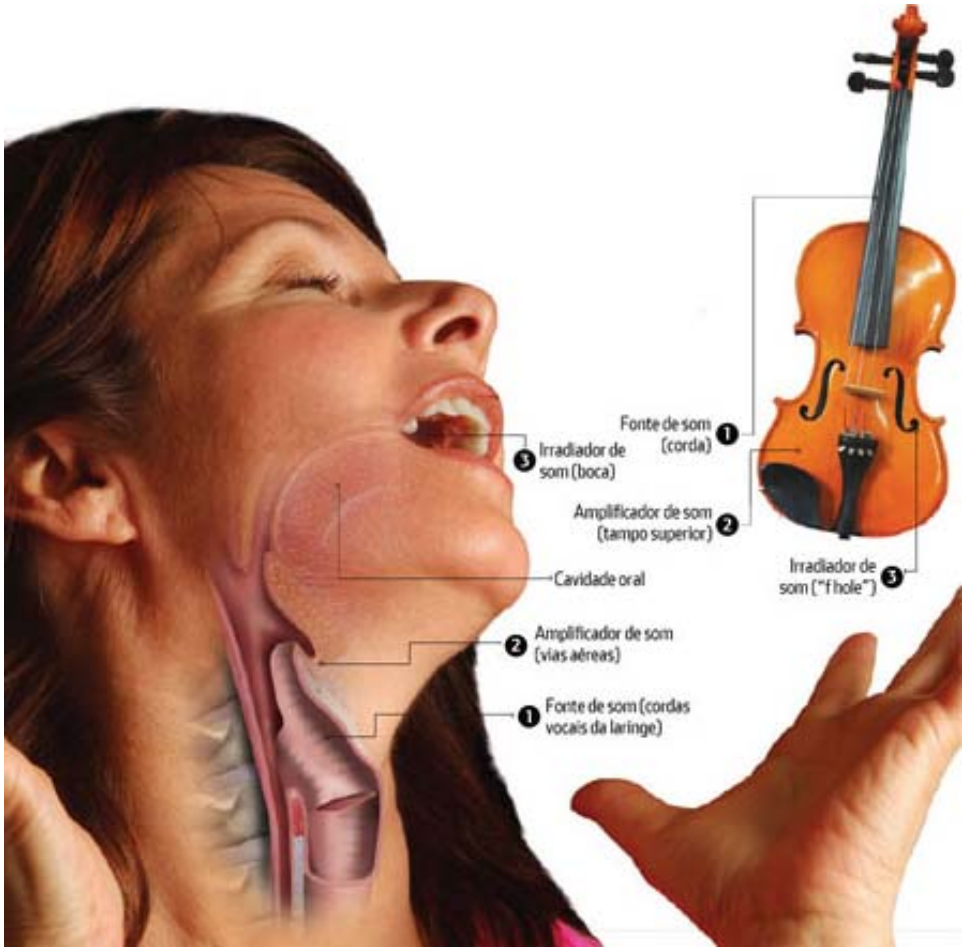

Figura 4 – Produção do som
Fonte: LEMOS, 2014.

Onde 1 representa o vibrador, 2 os ressonadores, e 3 os articuladores. (CUERVO, 2010).

2.2 Síntese de Fala

A síntese de fala, implementada por sistemas tecnicamente conhecido como TTS (*Text-To-Speech*), é a produção artificial de uma voz a fim de imitar a fala humana. (ZUFFO; PISTORI, 2004). Ao contrário do reconhecimento de voz, o sintetizador recebe como entrada um texto escrito, e retorna o mesmo texto falado. Para a realização da síntese, os sintetizadores utilizam-se de quatro métodos normalmente: (ALBULQUERQUE, 2001).

- Síntese Articulatória;
- Síntese por Formantes;
- Síntese por Concatenação;
- Síntese LPC (*Linear Predictive Coding*).

A síntese Articulatória tenta modelar os órgãos do trato vocal citados no item 2.1, assim como os movimentos por eles realizados. Cada item modelado passa a ser um parâmetro de controle para sintetizadores articuladores; abertura dos lábios, protuberância dos lábios, abertura da glote, tensão das cordas vocais, pressão pulmonar e etc. (KRÖGER, 1992 apud ALBULQUERQUE, 2001).

O Waseda Talker é um projeto desenvolvido no laboratório Takanishi na Universidade Waseda no Japão que visa desenvolver um robô falante utilizando o método de síntese Articulatória. O som não vem de nenhum alto-falante ou sintetizador, mas sim de cordas vocais de silicone e aparatos que simulam órgãos humanos, dos dentes à língua, passando pelo nariz. No decorrer de quinze anos, sete versões já foram criadas, onde, algumas delas foram aperfeiçoadas. (MORI, 2008), (TAKANISHI LAB, 2015).

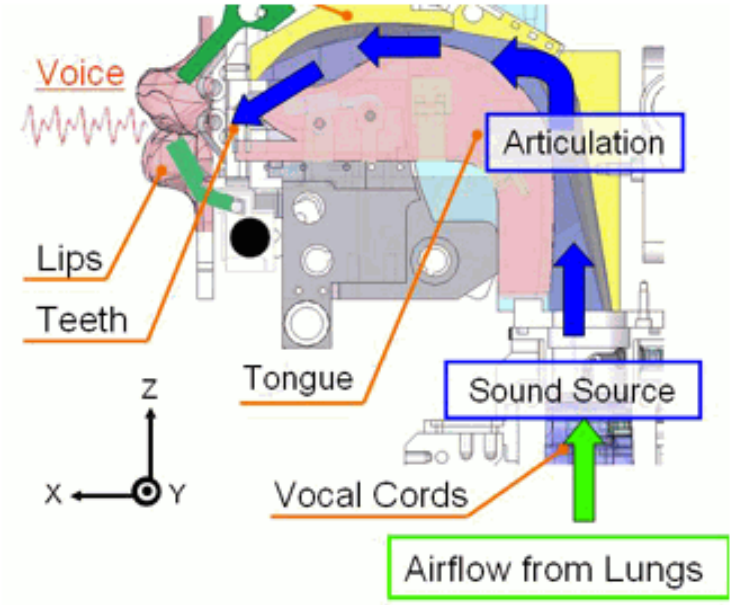

Figura 5 – Mecanismo do Waseda Talker
Fonte: MORI, 2008.

A síntese por Formantes modela um sinal fornecido por um fonte de excitação, com um filtro digital que implementa uma função de transferência, responsável por simular o processo de filtragem acústica existente no trato vocal. (HENTZ, 2009), (NAGLE; CHIQUITO, 1993), (SIMÕES, 1999).

Ao estabelecer-se um modelo adequado ao sintetizador, pode-se supor que ele é capaz de produzir uma fala na saída. Em outras palavras, o sinal sintético

gerado, depende da modelagem correta do processo de filtragem e também da fonte de excitação. (SIMÕES, 1999).

O sintetizador de Klatt é um sintetizador por Formantes baseado na teoria de produção de fala acústica desenvolvido por Fant (1960). (KLATT 1980, ALLEN et al. 1987, KLATT e KLATT, 1990 apud NAGLE; CHIQUITO, 1993). O processo é representado por um sistema linear, que simula as cavidades supralaríngeas com formatos e características variáveis no tempo, onde, ao receber sinais acústicos, filtra-os e emite ao meio externo, de maneira semelhante ao aparelho fonador. Sendo assim, este modelo é o resultado da combinação linear de três componentes básicos: i/ fontes sonoras, ii/ características de filtragem do trato vocal e iii/ características de radiação para o meio externo. (NAGLE; CHIQUITO, 1993)

Já a síntese por Concatenação, diferente da Articulatória e por Formantes, como o próprio nome induz, a concatena amostras pré-gravadas de unidades, podendo estas serem palavras, sílabas, fonemas, difones e trifones. (ALBUQUERQUE, 2001).

A palavra, de modo geral, é a unidade mais natural ao texto escrito, e sua concatenação é relativamente fácil de ser executada. Porém, há uma grande diferença de palavras faladas isoladamente, de palavras faladas dentro de uma frase, pois o som destas pode variar de acordo com sua posição. Outra resalva, se dá pela grande quantidade de palavras diferentes e nomes próprios, presentes em cada idioma. Assim, a palavra não é uma unidade satisfatória para um sistema de síntese de fala irrestrito. (ALLEN, 1987 apud ALBUQUERQUE, 2001).

As sílabas, por sua vez, apesar de conter um número menor de unidades, ainda é uma unidade insatisfatória, pois, diferentemente das palavras, os efeitos de pronúncia e entonação não são armazenados por unidades. Logo, não existem fundamentos técnicos para que se implemente sistemas TTS com unidades de palavras e sílabas. (ALBUQUERQUE, 2001).

Atualmente, os sistemas TTS estão, principalmente, baseados no uso de fonemas (Figura 6), difones, trifones ou a combinação dos mesmos, pois representam a linguística normal da fala. A quantidade de unidades básicas está entre 40 (quarenta) e 50 (cinquenta), que é claramente menor se comparada a

outros tipos de unidades; palavra e sílabas. (ALLEN, 1987 apud ALBUQUERQUE, 2001).

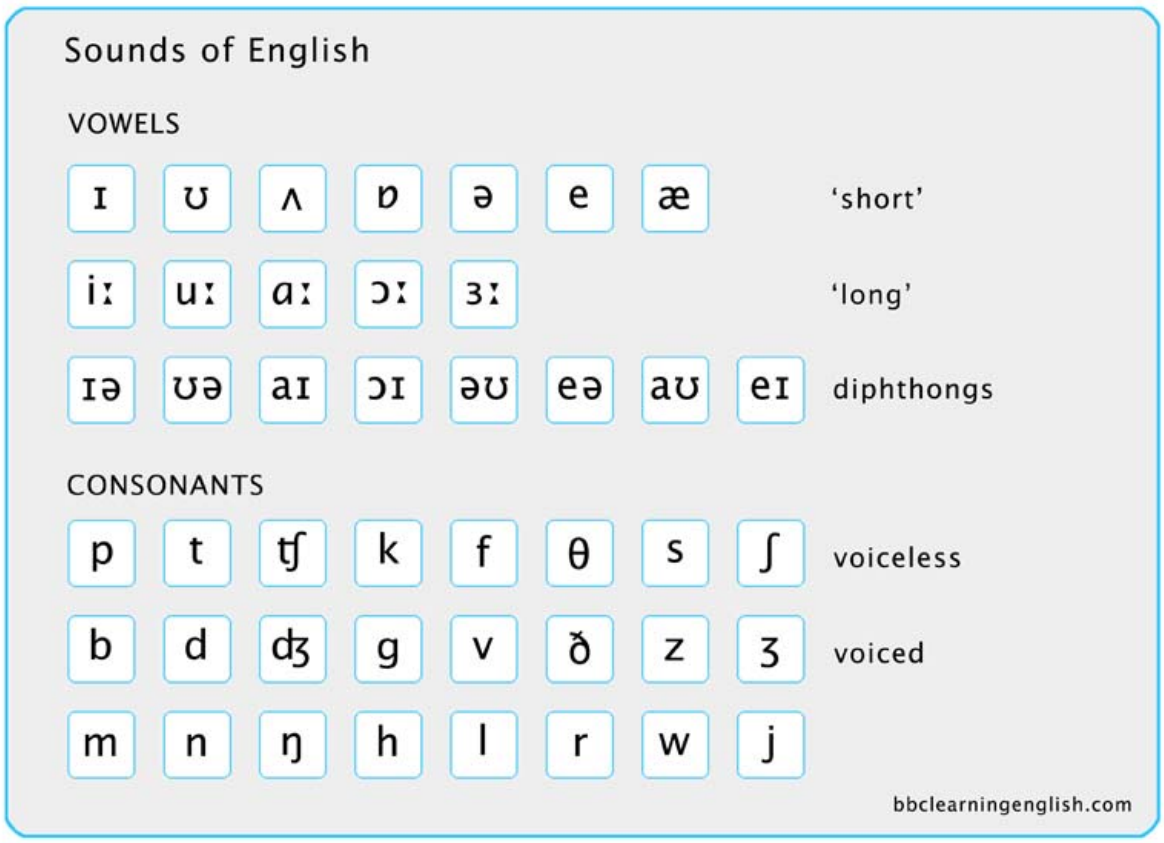

Figura 6 – Exemplo de fonemas para Inglês
Fonte: GUSMÃO, 2013.

O uso de fonemas fornece maior flexibilidade aos sistemas TTS, pois qualquer palavra, de qualquer idioma, pode ser formada a partir da concatenação de fonemas. Entretanto, alguns sons que não têm uma posição firmemente designada, como os sons explosivos, são difíceis de sintetizar. Sendo assim a articulação destes é formulada como regra. Outra desvantagem é a distorção existente entre os pontos de concatenação. Para remediar esse problema, utilizam-se difones, pares de fonemas, para reduzir a quantidade de concatenação e melhor a qualidade da síntese. Como os difones são as combinações existentes dos fonemas, o número de unidades aumenta de 40 (quarenta) a 50 (cinquenta), para 1500 (hum mil e quinhentos) a 2000 (dois mil). (ALBUQUERQUE, 2001).

Para suavizar a distorção, não só difones são utilizados; existem métodos como, PSOLA, TD-PSOLA e MBROLA, que controlam a duração e o tom de cada fonema. (ALBUQUERQUE, 2001).

Unidades segmentárias maiores, como trifones ou tetrafones, são raramente utilizadas por aumentar exponencialmente a quantidade de unidades. (ALBUQUERQUE, 2001).

O método LPC é uma das técnicas de síntese de voz mais poderosa, e um dos métodos mais úteis para codificar a voz com boa qualidade, a uma baixa taxa de bit. Seu processo consiste em concatenação de formantes, modelados por uma função de transferência que simula o som dos fonemas, diferentemente do sintetizador de Klatt, que modela uma função de transferência para simular as características de ressonância do trato vocal. (ALBUQUERQUE, 2001), (NAGLE; CHIQUITO, 1993).

Programas TTS inteligíveis permitem que pessoas com deficiência visual, dificuldade de leitura ou fala (Figura 7 – Stephen Hawking), possam ouvir obras escritas em um computador em casa e/ou se comunicar livremente.

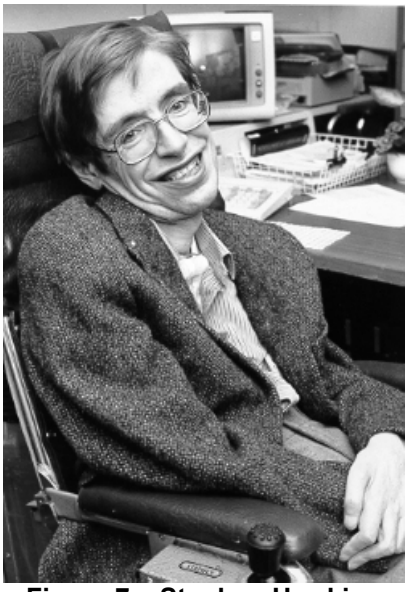

Figura 7 – Stephen Hawking
Fonte: NASA Star Child, [19--?].

2.3 GPS

Através de sinais emitidos por 24 satélites GPS que circulam a Terra, duas vezes ao dia, em orbitas bem precisas (Figura 8), é possível determinar qual a posição exata na Terra, de qualquer dispositivo que consiga receber dados dos mesmos. (GABRIEL, 2014).

Figura 8 – Sistema de Satélites GPS
Fonte: GARMIN, 2015.

O sistema de GPS foi criado pelo Departamento de Segurança americano no projeto inicialmente chamado de NAVSTAR. O objetivo desse projeto era desenvolver um sistema que disponibilizaria, várias informações geográficas de qualquer parte da superfície terrestre. O projeto NAVSTAR foi iniciado em 1960, e após vários anos de correções e ajustes, tornou-se totalmente funcional e pronto para operar somente em 1995. (GABRIEL, 2014).

Com o sucesso evidente, o projeto teve o nome alterado para GPS - *Global Positioning System* – em português, Sistema de Posicionamento Global. Outra mudança ocorreu quando o presidente Bill Clinton, em 2000, decidiu tornar as informações transmitidas pelos satélites, disponíveis gratuitamente para o uso civil; um sistema que a princípio era de uso exclusivo militar. (GABRIEL, 2014).

Os envios dos sinais são realizados ininterruptamente por todos os satélites, e de maneira simultânea, os receptores recebem os sinais, que por triangulação calculam a posição do usuário com boa precisão. (GARMIN, 2015).

Essencialmente, o receptor compara o tempo em que o sinal foi transmitido pelo satélite, com o tempo que foi recebido pelo receptor, sendo que esta diferença informa ao receptor qual a distância entre ele e o satélite. Com este referencial de pelo menos três satélites, o receptor consegue determinar sua localização, $\{x, y, z\}$,utilizando o conceito de triangulação com os dados recebidos - três variáveis: três equações. Com dois satélites, a posição fica imprecisa por existir duas possíveis localizações (Figura 9). Com quatro satélites ou

mais, é também possível sincronizar o tempo do dispositivo receptor com o relógio de alta precisão disponível nos satélites. (MARTINS, 2009), (GARMIN, 2015).

Figura 9 – Triangulação com três e dois satélites
Fonte: MARTINS, 2009.

Os satélites GPS transmitem dois sinais de rádio de baixa potência, designados L1 e L2. GPS Civil usa apenas a frequência L1, enquanto o GPS Militar utiliza ambas. L1 e L2 enviam o mesmo sinal, sendo assim não existe diferença de precisão entre os dois tipos de GPS. No entanto, como GPS Militar recebe em duas frequências, eles podem ser comparados e corrigidos, reduzindo assim a degradação do sinal. (GARMIN, 2015), (GPS.Gov, 2015).

2.4 Metodologia de Desenvolvimento

Considera-se um projeto bem-sucedido, quando ao seu término, ele atingir as necessidades dos usuários finais. Porém, ele será ótimo se além de eficiente, ele tiver sido projetado de maneira eficaz; seguindo cronograma e orçamento previamente definidos. (FALBO, 2005).

Projetos com um fluxo de desenvolvimento casual, raramente conseguem seguir um calendário. Assim, diversas metodologias emergiram a fim de facilitar a conclusão de projetos ótimos, onde as características de cada projeto define qual é o método mais apropriado. (FALBO, 2005).

Os métodos se dividem entre: i/ processo de desenvolvimento sequencial e ii/ processo de desenvolvimento evolutivo. Entre os modelos mais conhecidos estão o Modelo em Cascata e o Modelo em Espiral para cada método, respectivamente. (TRUYTS et al., 2012).

Nos modelos sequenciais, uma tarefa só é iniciada após o término da atividade anterior. O Modelo em Cascata é o mais simples dentre os sequenciais, e utiliza a sequência clássica como fluxo de atividades: (ROYCE,1970):

1. Análise e definição de requisitos;
2. Planejamento do sistema;
3. Implementação;
4. Teste;
5. Execução em produção;
6. Manutenção;

Esse modelo geralmente é utilizado para sistemas menores, onde os requisitos estão bem difundidos na equipe do projeto. (FALBO, 2005).

No entanto, este modelo tem sido considerado antiquado e simplista pelos defensores do projeto orientado a objeto, que na maioria das vezes utilizam o Modelo em Espiral (FOLDOC, 2014).

O modelo Espiral é um processo de desenvolvimento evolutivo, que tem como característica principal a divisão em ciclos. Em cada ciclo é realizado uma sequência do Modelo em Cascata. A maior vantagem desse modelo é a detecção antecipada de problemas, onde soluções são facilmente implementadas. Deve-se optar por esse modelo quando o problema não é bem definido e ele não pode ser totalmente especificado no início do desenvolvimento; também recomenda-se para projetos de grande escala. (FOLDOC, 2014) (FALBO, 2005).

Para definir qual modelo é o mais adequado, é necessário verificar algumas das características referentes ao projeto (FALBO, 2005):

- Tamanho e complexidade do sistema;
- Estabilidade dos requisitos;
- Características da equipe;

Essa definição deve ser o ponto de partida para o desenvolvimento de um projeto ótimo. (FALBO, 2005).

O SiNDi é um projeto pequeno, de complexidade razoável devido aos elementos de software e hardware envolvidos, onde o requisito do cliente é bem específico, sendo definido no item 1.3.1. A equipe do projeto é formada por apenas

uma pessoa, possibilitando a utilização de modelos mais simplistas. Sendo assim suas características, apontam para o modelo sequencial.

Dentre as metodologias apresentadas para modelos sequencias, o Modelo-V (Figura 10) foi a selecionada pois ele procura enfatizar e estreitar a relação entre as atividades de teste (teste de unidade, teste de integração, teste de sistema e teste de aceitação) e as demais fases do processo. (FALBO, 2005).

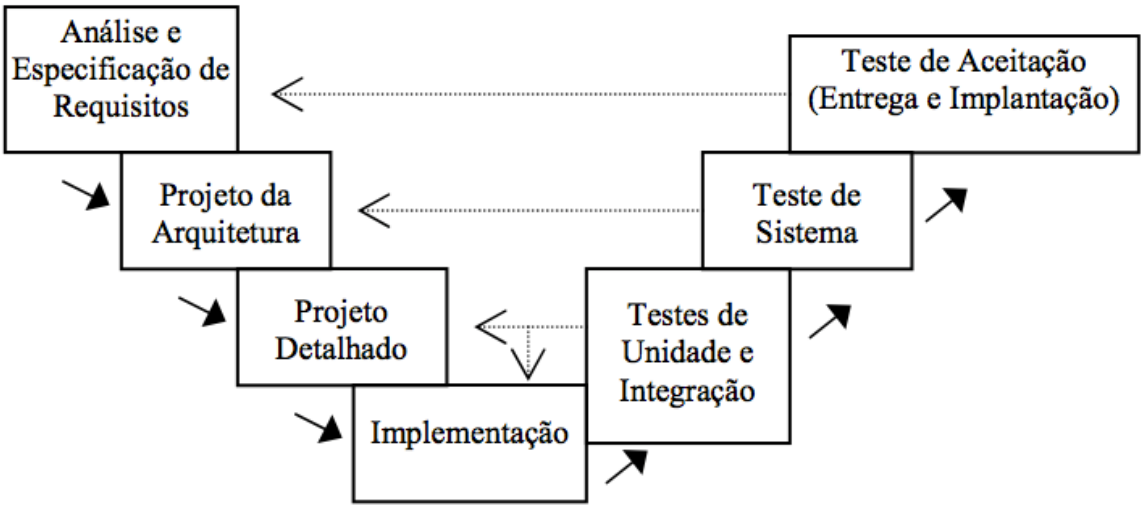

Figura 10 – Modelo em V
Fonte: FALBO, 2005.

3 METODOLOGIA

De acordo com o modelo de processo escolhido, foram determinadas as atividades e subatividades à serem seguidas no desenvolvimento do projeto:

Quadro 1 – Sequência de atividades do projeto

Projeto
Análise e Especificação de Requisitos
Especificação de requisitos pela URBS
Definição de solução
Análise de tecnologias
Escolha de tecnologia
Especificação de Teste de Aceitação
Projeto de Arquitetura
Diagrama de blocos
Diagrama de sequência
Especificação de Teste de Sistema
Projeto Detalhado
Sistema Embarcado
Diagrama de estados
Diagrama de tarefas
Estrutura de mensagens enviadas
Tratamento de mensagens
Central
Diagrama de casos de uso
Diagrama entidade-relacionamento
Dicionário de Dados
Especificação de Testes de Unidades
Especificação de Testes de Integração
Análise de Riscos
Implementação
Sistema Embarcado
Festival C++
Cliente/Servidor C++
Protocolo de Comunicação com Interpretador de mensagem
Escalonador por tempo C++
Escalonador por coordenadas GPS C++
Central
Banco de Dados da Central SQL
Software para Central C#
Cliente C#
Teste de Unidade (Ambiente de Desenvolvimento)
Sistema Embarcado
Festival C++
Cliente/Servidor C++

Fonte: Autoria Própria, 2014.

Quadro 2 – (Continuação) Sequência de atividades do projeto

Escalonador por coordenadas GPS C++
Central
Banco de Dados da Central SQL
Software para Central C#
Cliente C#
Implementação
Correções sobre os testes de unidades falhados
Teste de Integração (Ambiente de Desenvolvimento)
Sistema Embarcado
Festival e Escalonador por tempo
Festival, Escalonador por tempo, Cliente/Servidor C++
Festival e Escalonador por coordenadas GPS
Festival, Escalonador por coordenadas GPS e Cliente/Servidor C++
Festival, Escalonador por tempo e coordenadas GPS e Cliente/Servidor C++
Central
Banco de Dados e software usuário C#
Banco de Dados, software usuário, Cliente C#
Teste de Sistema (Ambiente de Desenvolvimento)
Teste de Aceitação (Ambiente de Aceitação)

Fonte: Autoria Própria, 2014.

4 PROJETO

4.1 Análise de Requisitos

4.1.1 Problema e Necessidade

O problema relatado pela URBS, apresenta a insatisfação com o processo de atualização das áudio-mensagens nos ônibus, que leva em torno de 10 dias, sendo a duração consequência dos elementos no qual o processo depende: de locutor e recurso para se locomover até a garagem e trocar a memória no ônibus. Deste modo foi explicitada a necessidade de eliminar a dependência dos recursos no processo.

4.1.2 Solução

Como solução, o projeto aborda a utilização de síntese de fala para substituir o locutor e um sistema com conexão remota para alteração das mensagens via *wireless* eliminando a necessidade de ir até o local onde o ônibus se encontra.

4.1.3 Requisitos

De acordo com a oportunidade e solução definida são requisitos necessários:

- Programa de síntese de fala que permita implementação em plataforma *open source*;
- Programa de síntese de fala que apresente uma qualidade aceitável do voz sintetizada;
- Programa de síntese de fala que permita a criação de nova voz;
- Programa de síntese de fala com documentação abrangente para implementação do projeto;
- Programa de síntese de fala que permita a integração em software a ser desenvolvido;
- Computador com plataforma de desenvolvimento *open source*;
- Computador para utilização embarcado que suporte o programa escolhido de síntese de fala;
- Protocolo de comunicação que seja compatível com transmissão sem fio;

4.1.4 Casos de Uso

O caso de uso é uma técnica de modelagem UML de requisitos que descreve o que um sistema é capaz de fazer, no qual devem (ECLIPSE, 2015):

- descrever como os usuários interagem com o sistema;
- as funcionalidades do sistema, dando uma visão externa aos clientes;

Sumarizando, eles descrevem o que o sistema faz, mas não especificam como será feito.

O SiNDi apresenta o diagrama de casos de uso conforme apresentado pela Figura 11.

Aqui são apresentadas as funcionalidades da Central e do Sistema Embarcado através dos seus respectivos atores, no qual suas relações serão explicadas nos itens ao longo do trabalho.

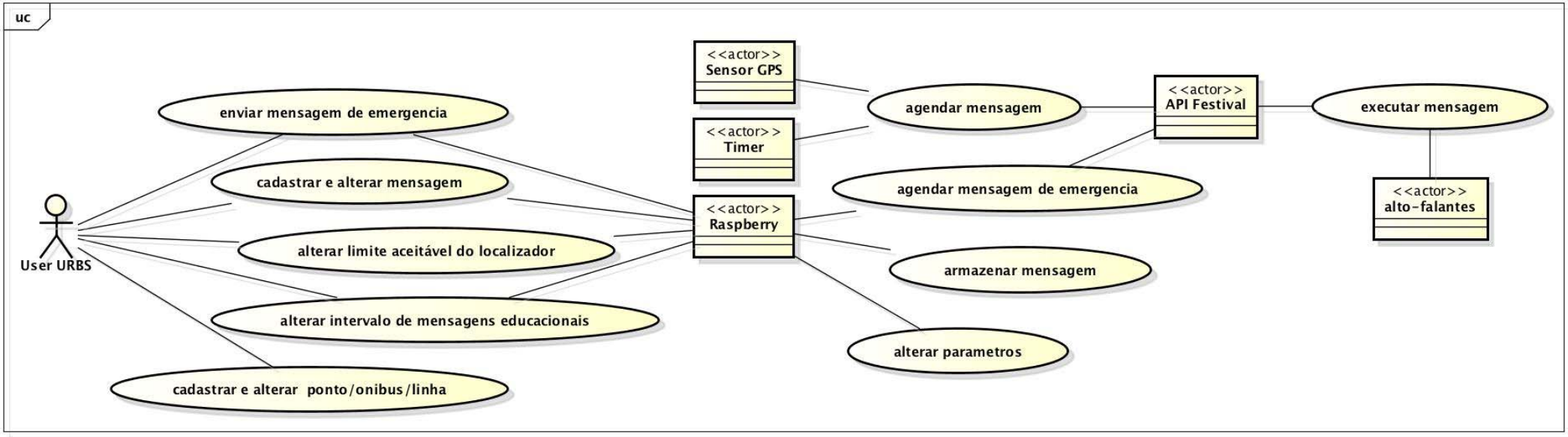

Figura 11 – Diagrama de Casos de Uso
Fonte: Autoria Própria, 2014.

4.1.5 Análise de Riscos

Para analisar os riscos, foi definida primeiramente a classificação para impacto, probabilidade (Quadro 3) e pertinência (Quadro 4), no qual o último é resultado da relação entre os dois primeiros.

Quadro 3 – Classificação dos níveis de impacto e probabilidade

0 - 20%	Muito Baixo	MB
21% - 40%	Baixo	B
41% - 60%	Moderado	M
61% - 80%	Alto	A
81% - 100%	Muito Alto	MA

Fonte: Autoria Própria, 2014.

Quadro 4 – Quadro de pertinência por impacto e probabilidade

		Impacto				
		MB	**B**	**M**	**A**	**MA**
Probabilidade	**MB**	MB	MB	MB	B	B
	B	MB	B	B	M	M
	M	MB	B	M	A	A
	A	B	M	A	A	MA
	MA	B	M	A	MA	MA

Fonte: Autoria Própria, 2014.

Os riscos levantados no projeto são apresentados no Quadro 5.

Quadro 5 – Riscos do projeto

Código do Risco	Risco	Probabilidade	Impacto	Pertinência
1	Perda de Software	MB	MA	B
2	Problemas com Hardware	MB	M	MB
3	Problemas de conexão	A	M	A

Fonte: Autoria Própria, 2014.

Riscos com pertinência MB, ou B foram desconsiderados sobre o efeito da convivência. Logo o risco 3 é o único que deve ser descrito.

Existem duas conexões necessárias no projeto, que podem apresentar problemas de transmissão de dados: i/ Central-SE e iii/ satélite-módulo GPS. Todas eles utilizam protocolos que implementam o conceito ACID para as transições - Atomicidade, Consistência, Isolamento e Durabilidade – logo, essas propriedades eliminam riscos de transmissão.

Entretanto, quando a conexão for perdida, não há como receber ou enviar sinal. Para mitigar os efeitos da perda de conexões, a implementação foi estruturada

para que esse tipo de falha não atrapalhe na execução das áudio-mensagens contidas no SE.

Como as mensagens são armazenadas no SE através de uma estrutura, a comunicação Central-SE não precisa ser mantida a todo o momento. A conexão só é aberta, quando a Central deseja enviar um pacote, e que neste caso, se a conexão falhar no momento do envio, o programa ficará tentando até conseguir se comunicar.

Para mitigar o problema de conexão com o satélite da rede GPS, o programa é configurado para sempre manter a última coordenada enviada pelo módulo salvo, a fim de que se houver uma falha na conexão, a posição do ônibus seja no lugar mais antigo registrado.

4.1.6 Especificação do Teste de Aceitação e Sistema

O teste de aceitação é o último antes de levar o sistema para produção. Por isso, a situação de teste deve ser similar ao ambiente real; o teste deve ser conduzido pelo cliente, onde cenários não definidos serão testados aleatoriamente. (ROCHA, 2011).

Neste momento, o cliente tem o intuito de observar se os requisitos foram atendidos e se sistema funciona adequadamente, a fim de aceitar ou recusar o sistema. (ROCHA, 2011).

O teste de sistema é o último teste realizado pelo desenvolvedor. Aqui deve-se testar todas as funcionalidades disponíveis no sistema, em busca de falhas por meio da utilização do mesmo, como se o desenvolvedor fosse um usuário final. O ambiente, assim como no teste de aceitação, deverá ser o mais próximo possível ao ambiente de produção. (ROCHA, 2011).

Os testes de sistema e aceitação são parecidos, porém um é realizado pelo usuário final e outro pelo desenvolvedor. (ROCHA, 2011).

O SiNDi sendo um projeto universitário e, inicialmente, sem cunho econômico, possui uma interação branda com o cliente. Portanto, foi decidido pela suspensão do teste de aceitação, e definido cenários que simulariam o comportamento do usuário no teste de sistema.

Os cenários são estes apresentados a seguir:

I. Envio de três mensagens educativas;
   - “Cuidado com furtos no interior do veículo.”;
   - “Esta linha estará inoperante no domingo, dia 14 de dezembro, das 6 às 10 horas da manhã, devido a Caminhada do Coração.”
   - “Aguarde sempre o desembarque.”

Resultado esperado: a cada 10 minutos a mensagem com o horário de execução mais antigo, deverá ser executada no SE.

II. Envio de duas mensagens por localização;
   a. “Próxima parada, estação praça Eufrásio Correia. Desembarque por todas as portas.”
   b. “Próxima parada, estação Alferes Poli. Desembarque pelas portas 2 e 4.”

Resultado esperado: ao se aproximar das estações praça Eufrásio Correia e Alferes Poli, a mensagem correspondente deverá ser executada no SE.

III. Alteração do intervalo de mensagens educativas;

Resultado esperado: a cada *t* minutos a mensagem educacional com o horário de execução mais antigo, deverá ser executada no SE.

IV. Reinicialização de mensagens para uma linha;

Resultado esperado: até que todas as mensagens referente a linha sejam recebidas, o SE permanece em silêncio. Após o término no envio, o sistema deverá se comportar de acordo com os itens II e III.

4.2 Análise de Tecnologias

4.2.1 Síntese de Fala

Grande nomes da informática como Microsoft e Google, já fornecem bibliotecas de sistemas TTS para serem utilizadas em suas plataformas. Em ambiente Windows, o SDK Microsoft Speech Platform, por exemplo, é uma ferramenta poderosa, que permite ao desenvolvedor utilizar reconhecimento de voz e sistemas TTS em suas aplicações. (MICROSOFT, 2015).

Para o SiNDi, foram analisados três sintetizadores que disponibilizam implementação em plataforma aberta:

- Festival: síntese por concatenação; contendo palavras, sílabas e

fonemas. (TAYLOR; BLACK; CALEY, 1998);

- e-Speak: síntese por Formantes; pode ser usado também com síntese por concatenação de difones. (SourceForge.NET, 2015);
- fLite: síntese por concatenação de difones. (BLACK; LENZO, 2001).

O primeiro item analisado foi a qualidade da sintetização. Nesse aspecto o Festival e o e-Speak se sobressaíram ao fLite sendo eliminado do leque de possibilidades.

O segundo item analisado foi a possibilidade de implementação de novas vozes, porém ambos, Festival e eSpeak, atenderam o requisito.

O terceiro e decisivo item a ser analisado para a escolha foi a qualidade da documentação apresentada pelos desenvolvedores. A documentação disponível para o Festival, é muito superior aos demais, possuindo exemplos, explicação de bibliotecas, processos de implementação e etc.

Sendo assim o Festival foi o programa escolhido para realizar a síntese de fala do SiNDi.

4.2.2 Computador para projetos embarcados

Atualmente, existem diversas possibilidades de sistemas embarcados para criação de projetos. Entre os mais conhecidos estão Arduino e Raspberry Pi. Ambos são plataformas de desenvolvimento *open source*, porém uma é composta por um microcontrolador e outra por um microprocessador, respectivamente. (ORSINI, 2014).

Em geral, um microprocessador é apenas a CPU, enquanto os microcontroladores possuem todos os módulos da arquitetura Von-Neumann, memórias e E/S, acoplados. Assim, é correto afirmar que os microcontroladores, contém em seu interior um microprocessador (Figura 12). (SILVA; CÂNDIDO, 2011).

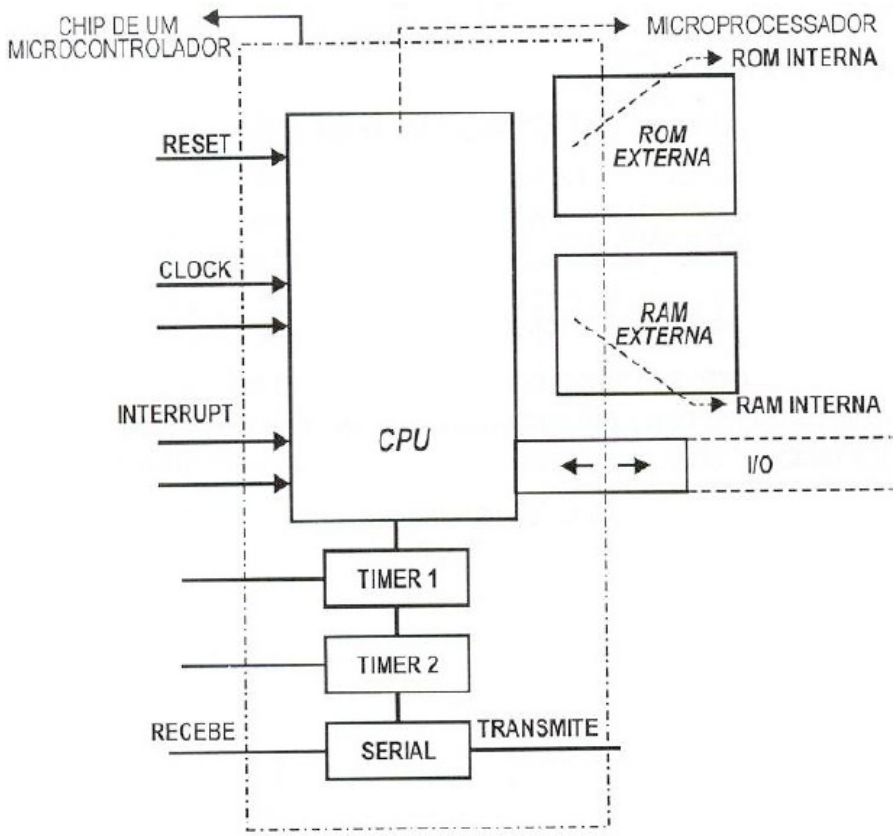

Figura 12 – Microcontrolador contém um microprocessador
Fonte: SILVA; CÂNDIDO, 2011.

Entretanto, por um microprocessador ser apenas a CPU, ele possui mais capacidade de processamento do que a CPU de um microcontrolador. Com isso, acaba-se por decidir pelos microprocessadores, quando os projetos são para o alto processamento, trabalhando com grandes quantidades de dados em grande velocidade, e pelos microcontroladores quando trabalham com poucos dados a uma baixa frequência. (SILVA; CÂNDIDO, 2011). A figura a seguir, através da comparação entre o Arduino e o Raspberry, apresenta essa diferença em números.

Quadro 6 – Arduino vs Raspberry Pi

	Arduino	**Raspberry Pi Model B**
Preço	$ 30.00	$ 35.00
Tamanho	7.6 x 1.9 x 6.4 cm	8.6 x 1.7 x 5.4 cm
Memória	0.002 MB	512 MB
Velocidade do Clock	16 MHz	700 MHz
Placa de Rede Acoplada	Não	Sim
Multitarefa	Não	Sim
Tensão de Entrada	7-12 V	5 V
Memória Flash	32 KB	SD Card (2-16G)
USB	uma, somente entrada	duas
Sistema Operacional	Nenhum	Linux
IDE	Arduino	qualquer IDE suportada pelo Linux

Fonte: ORSINI, 2014.

O SiNDi é um sistema cuja sua principal função é a síntese de fala, que requer alta capacidade de processamento. Desta apresentada, foi decidido pela utilização do Raspberry PI por possuir maior capacidade de processamento ao ser comparado com Arduino.

4.2.3 Banco de Dados

Do bancos de dados existentes, o SQL server foi o escolhido devido à facilidade e à familiaridade com a ferramenta e ambiente de desenvolvimento Visual Studio.

4.3 Arquitetura do Sistema

O SiNDi é um sistema composto por uma Central, um sistema embarcado (SE) e um sistema de comunicação entre estes dois elementos.

A Central fica localizada na URBS e é caracterizada por um computador com o software de manipulação de mensagens instalado, utilizado por funcionários da URBS.

O SE é móvel e fica localizado nos ônibus. Ele é caracterizado por possuir um sensor GPS e alto-falantes, acoplados a um hardware configurado para se comportar como um servidor embarcado na manipulação das mensagens enviadas pela Central.

O sistema de comunicação, Central – SE, utiliza comunicação baseada em sockets TCP/IP com trocas de mensagens do tipo texto.

A Figura 13 apresenta o diagrama do SiNDi.

A Central permitirá ao funcionário da URBS modificar nos ônibus as notificações existentes e suas características, remotamente.

O SE, ao lado direito, é responsável por armazenar, agendar e executar as áudio-mensagens através de interrupções de tempo e localização. O SE também pode realizar tarefas não espontâneas determinadas e enviadas pela Central.

As interrupções por tempo ocorrem a cada 10 minutos, no qual é escalonado para execução a mensagem que possui seu horário de última execução mais antigo.

Já o escalonamento por local, o usuário define o local onde a mensagem deve ser executada, e sempre que o GPS se encontra na posição desejada, o agendamento é realizado.

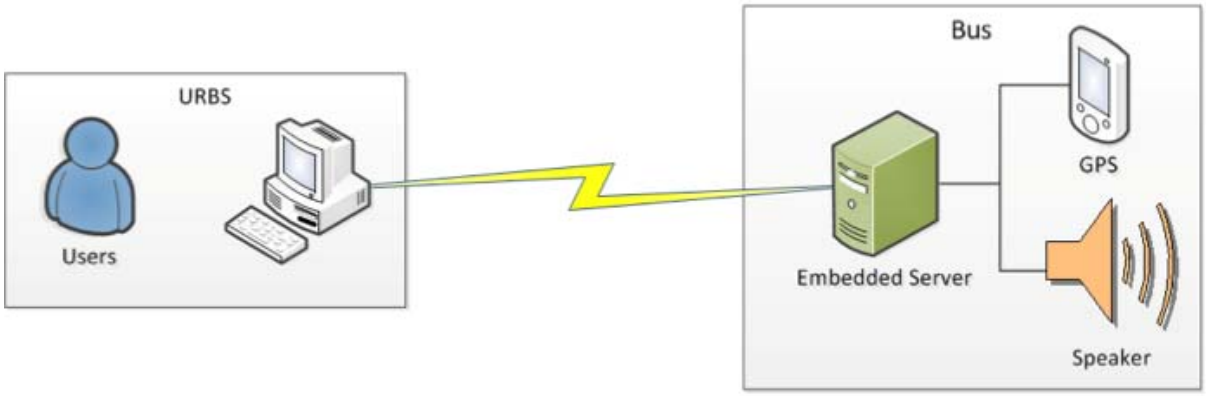

Figura 13 – Diagrama do Projeto
Fonte: Autoria Própria, 2014

Em eletrônica, os diagramas de blocos são utilizados para apresentar módulos e/ou elementos físicos presente no sistema, pois conseguem transmitir com clareza a relação entre os mesmos.

Atualmente, não existe documentado nenhuma padronização para a utilização do diagrama de blocos, assim sua utilização é livre de qualquer especificação.

No SiNDi tem-se 6 elementos distribuídos como apresentado na Figura 14:

- Computador (Central);
- Módulo de rede sem fio;
    - WI-FI ou 3G;
- Módulo GPS;

- Antena externa GPS;
- Alto-falantes;
- Raspberry PI;

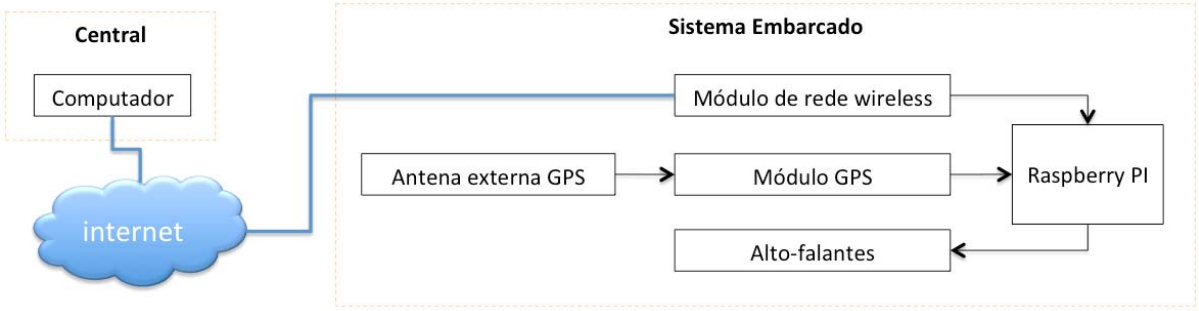

Figura 14 – Diagrama de Blocos
Fonte: Autoria Própria, 2014

A Central e o SE se comunicam através de tramissão de pacotes por uma rede, podendo ser privada limitada (*internet*), privada global (intranet), pública global (Internet).

Para a Central não é necessário a especificação do meio que o computador se conectará à rede; cabo ethernet ou sem fio. É comum que uma das duas tecnologias esteja presente nesse computador.

Para o SE, a conexão na rede deverá ser realizada via comunicação sem fio, pois um dos requisitos do cliente é remover do processo de atualização de mensagens, a necessidade de ir até o ônibus.

A implementação da interface de comunicação sem fio não faz parte do escopo do projeto, assim implementou-se o protocolo de comunicação TCP/IP para que, posteriormente, qualquer tecnologia de conexão wireless com o mesmo protocolo fosse realizada facilmente.

O módulo GPS comunica-se comunicação serial com o Raspberry, podendo a conexão ser realizada por pinos ou uma das portas USB. Para conectar a antena externa a esse módulo é necessário um adaptador µFL para SMA, pois a antena tem entrada SMA e o módulo oferece saída µFL.

Os alto-falantes são conectados na saída de áudio analógico e estéreo do Raspberry.

4.3.1 Arquitetura Servidor – Sistema Embarcado

De acordo com as características já exploradas nos capítulos anteriores, observou-se que existem atividades bem definidas para o SE, no qual não podem ser executadas em segundo plano. Sendo assim é nítida a necessidade de um programa que execute processos concorrentes, sendo eles descritos por:

A. Processamento de mensagens recebida da Central;
B. Sintetizador de voz com execução sequencial;
C. Temporizador, para interrupções de tempo;
D. Localizador, para interrupções de local;

Analisando-as é fácil compreender os efeitos, caso qualquer uma das ações bloquear a execução das demais tarefas.

Entretanto, todos os processos trabalharam com o mesmo conjunto de dados, sendo estes as mensagens enviadas pela Central. Neste caso, para que seja mantida a integridade dos dados, o conceito de semáforo deverá ser adotado nas estruturas que forem utilizadas por mais de um processo.

Desse modo, já se torna eminente, que o conjunto de dados citado, seja apresentado como uma estrutura que concentre a lista de todas as mensagens configuradas para aquele ônibus, assim como suas características.

Para isso é necessária a criação de uma estrutura de mensagem que acople todos os elementos da mensagem; um vetor do tipo mensagem será suficiente para realizar a função de armazenamento de mensagens vindas da Central.

Contudo, ainda que os processos trabalhem com a mesma estrutura, para uma implementação sofisticada, é necessário que cada processo tenha limites quanto às ações nas estruturas compartilhadas, portanto, o ideal é manter apenas um processo trabalhando com a estrutura do vetor, no que diz respeito a suas características gerais, e permitir aos demais processos apenas acesso para leitura nos elementos dos vetores.

A limitação acima é facilmente aceita, ao perceber que dentre os processos, realmente, apenas um deles trabalha com alterações estruturais e outros apenas necessitam acessar ou alterar algum item específico para algum elemento. Por exemplo: quando uma mensagem é recebida pela Central, o processo A se encarrega de tratar o pacote recebido, no qual pode estar apenas enviando uma

nova mensagem a ser executada por aquele ônibus. Nesse caso, o tratamento apropriado é simplesmente armazenar essa mensagem no vetor de mensagens existente para o ônibus. Essa é uma ação que altera a estrutura do vetor, pois aumentando o tamanho do vetor.

Porém, nos processos B, C e D não existe a necessidade de inserção de novos elementos, pois eles nem sequer possuem interface com a Central.

Assim sendo, definiu-se que:

- ◊ o processo B realiza a síntese de fala das mensagens e execução das mesmas nos ônibus, quando o critério das mensagens forem atendidos. Destaca-se nesse processo, que não é de sua responsabilidade reconhecer se o critério da mensagem já foi atendido. Resumindo ao ser enviado um pedido a ele, ele processa sem questionar;
- ◊ o processo C, por sua vez, é um dos responsáveis por “dizer” ao processo B o que ele deve executar, exclusivamente para mensagens do tipo educacional; a cada *t* minutos o processo gera uma interrupção que sequencialmente tem o objetivo de identificar qual das mensagens educativas deve ser falada;
- ◊ o processo D também é responsável por direcionar pedidos para o processo B, porém, ele procura por mensagens de localização, que possuam coordenadas aceitáveis, de acordo com o limite configurado.

O processo A, além de receber mensagens à serem executadas, também pode receber:

- ➢ mensagens de emergência, que devem ser executadas imediatamente desconsiderando as demais mensagens;
- ➢ pedidos de limpeza de mensagens no ônibus;
- ➢ pedidos de alteração de configurações.

Pelo descrito acima, foi adotada a estratégia composta por um vetor de mensagens, dois vetores de ponteiros de mensagens, três variáveis semáforo para os respectivos vetores, uma variável que define o intervalo em minutos das mensagens de educativas e uma variável que define o limite em metros aceitável para execução das mensagens de localização.

Isso definido, criou-se uma máquina de estados (Figura 15) para descrever o comportamento do software implementado no SE.

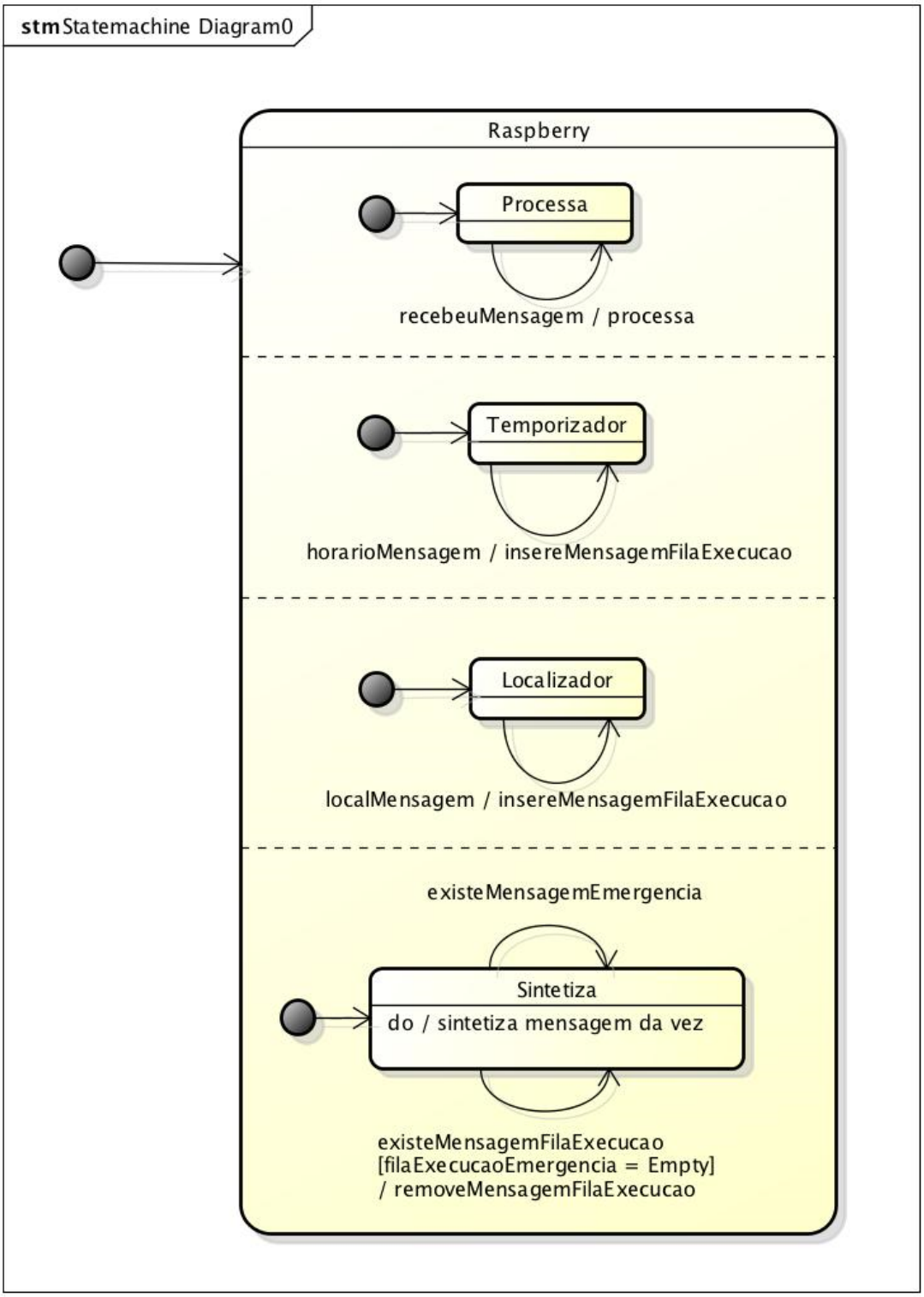

Figura 15 – Máquina de Estados
Fonte: Autoria Própria, 2014

São estados concorrentes: Processa, Temporizador, Localizador e Sintetiza; que representam respectivamente os processos A, C, D e B. Cada um dos estados aguardam um evento para executar uma ação.

Das ações apresentadas na máquina de estados, a única subjetiva é a processa, pois depende do tipo de mensagem que será recebida pelo servidor.

No diagrama de tarefas (Figura 16) é possível observar como os processos interagem com as estruturas definidas pelo projeto.

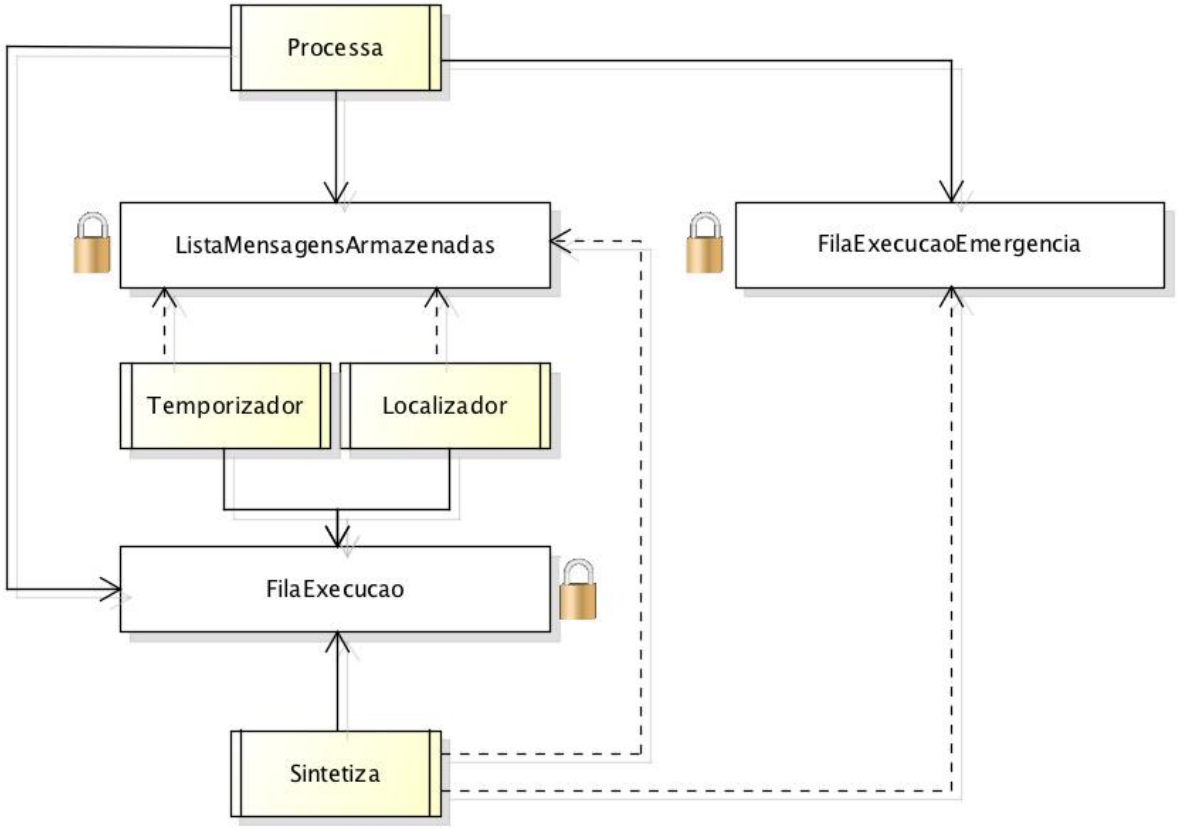

Figura 16 – Diagrama de Tarefas
Fonte: Autoria Própria, 2014

O Processa pode inserir e remover elementos de todos os vetores; de mensagens armazenadas, de mensagens de emergência e de mensagens agendadas para execução.

O Temporizador e o Localizador podem acessar e alterar elementos no vetor de mensagens armazenadas e inserir mensagens do vetor de execução.

O Sintetiza pode acessar e alterar elementos nos vetores de mensagens armazenadas e de emergência e remover mensagens do vetor execução.

Apesar do Temporizador/Localizador e Sintetiza realizarem ações de mudança de estrutura no mesmo vetor, elas são distintas pois uma apenas insere e outra apenas remove.

4.3.1.1 Especificação de Testes de Unidade

De acordo com a arquitetura apresentada e a metodologia apresentada no item 0, os testes devem ser feitos individualmente para cada elemento do Sistema Embarcado, sendo cada um descrito abaixo:

Escalonadores – Temporizador: através da determinação de um tempo fixo, declarado localmente, verificar se o temporizador reconhece a interrupção;

Escalonadores – Localizador: através da determinação de uma coordenada GPS fixa, declarada localmente, e um programa de simulação de ônibus que envia coordenadas GPS fictícias, verificar se o localizador reconhece quando o simulador está dentro do perímetro aceitável da coordenada GPS fixa.

Cliente/Servidor: verificar se o Cliente, definido como Central, envia mensagens para o servidor, sendo este o Raspberry.

Sintetizador: através da determinação de texto fixo, declarado localmente, verificar se o sintetizador realiza a síntese de fala da mensagem setada.

4.3.2 Arquitetura Cliente – Central

A Central não contém elementos complexos, pois seu objetivo é exclusivo para auxílio ao usuário.

Através da interface da Central, o usuário conseguirá configurar pontos, linhas, ônibus e mensagens. A Figura 17, apresenta todas as entidades e relacionamento presentes no BD (Banco de Dados) do SiNDi.

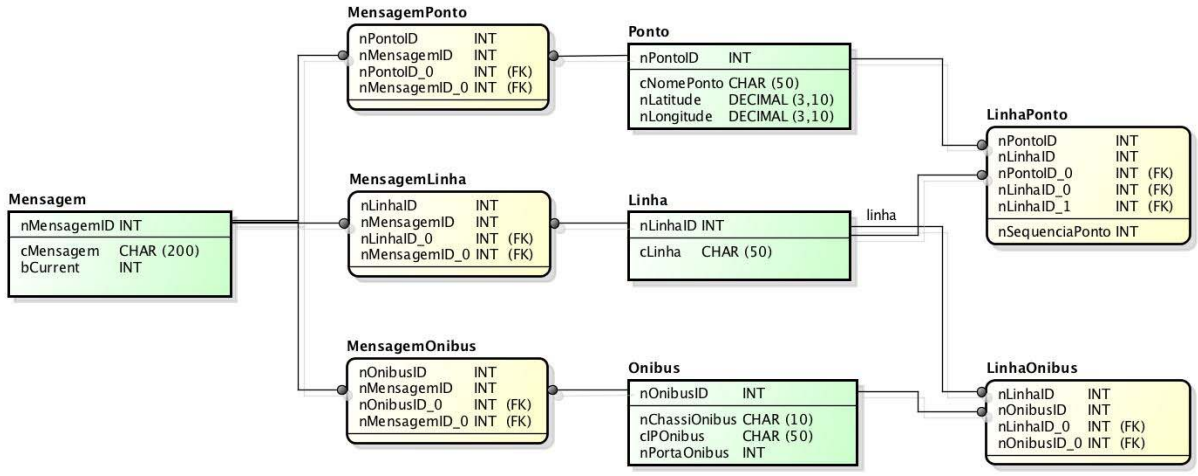

Figura 17 – Diagrama Entidade-Relacionamento
Fonte: Autoria Própria, 2014

Dos relacionamentos convencionais para RIT, destaca-se a relação linha-ponto-ônibus:

- uma linha pode conter *n* pontos e *n* ônibus;
- um ponto pode estar presente em *n* linhas;
- um ônibus pode operar em *n* linhas.

A relação ônibus-ponto é inexistente, pois ônibus não operam exclusivamente para um ponto.

Dos relacionamentos exclusivos do SiNDi, estão aqueles que envolvem a entidade *Mensagem*; uma mensagem pode estar relacionada ao um ponto, linha e/ou ônibus; o tipo das mensagens é definido dependendo da entidade com o qual ele se relaciona; mensagens relacionadas a *Ponto*, são do tipo *Localização*; mensagens relacionadas a *Linha* e *Ônibus* são do tipo *Tempo*;

Mensagens de emergência são definidas apenas na relação mensagem-ônibus, quando informado pelo usuário através da interface.

Para que sejam removidas as mensagens de emergência do ônibus, o usuário deve desabilitar as mensagens, cuja ação acarretará no envio pela Central ao SE, o pedido de limpeza do vetor de mensagens de emergência.

O envio de mensagens é realizado para o SE, no momento em que o usuário configura uma nova mensagem.

Para alterações de mensagens, a Central apenas envia o pacote para o SE, onde este, se encarrega de substituir as características alteradas;

Configurações do SE que podem ser alteradas via Central, não são necessárias de armazenamento em BD. Logo, são pacotes criados instantaneamente no momento em que o usuário realiza a requisição via interface.

4.3.2.1 Especificação de Testes de Unidade

O teste de unidade definido para a Central é simples, pois de acordo com o item 4.3, todo o processamento é realizado pelo servidor embarcado.

Sendo assim, deve-se verificar três aspectos: i/ funcionalidade do Banco de Dados, se está armazenando corretamente os dados inseridos pelo usuário, ii/ montagem da mensagem a ser enviada, se segue as sintaxes definidas no item

5.1.5 e iii/ transmissão da mensagem da Central para o Servidor, se está sendo realizada propriamente.

4.4 Especificação de Testes de Integração

Os testes de integração serão executados gradativamente em pares, conforme especificado na tabela de atividades apresentada no item 3.

5 IMPLEMENTAÇÃO

5.1 Sistema Embarcado

Devido à utilização de processos concorrentes, a codificação do SE foi implementada por *threads* em C++, no qual cada estado da Figura 15 no item 4.3.1 representa uma *thread*.

5.1.1 Escalonadores (Temporizador & Localizador)

A principal função do escalonador é agendar as mensagens que devem ser executadas pelo Festival. Assim toda a vez que é gerada uma interrupção por tempo, ou por localização, uma mensagem será reproduzida nos alto-falantes.

5.1.1.1 Por tempo C++ (Temporizador)

O escalonador por tempo, através de funções implementadas pela biblioteca de tempo padrão da linguagem C, time.h, gera interrupções a cada *t* minutos. O valor do intervalo é definido pela URBS e sempre que necessário pode ser alterado pela Central. Quando isto ocorre, são verificados os horários da última execução de todas as mensagens armazenadas pelo SE, aquela que possui o mais antigo.

5.1.1.2 Por coordenadas GPS C++ (Localizador)

Este escalonador tem como objetivo ler as coordenadas GPS fornecidas pelo módulo de GPS, e compará-las com as coordenadas das mensagens por localização armazenadas no SE. Ao encontrar uma mensagem dentro do perímetro, o escalonador gera uma interrupção para que a mensagem referente daquela localização seja sintetizada e executada.

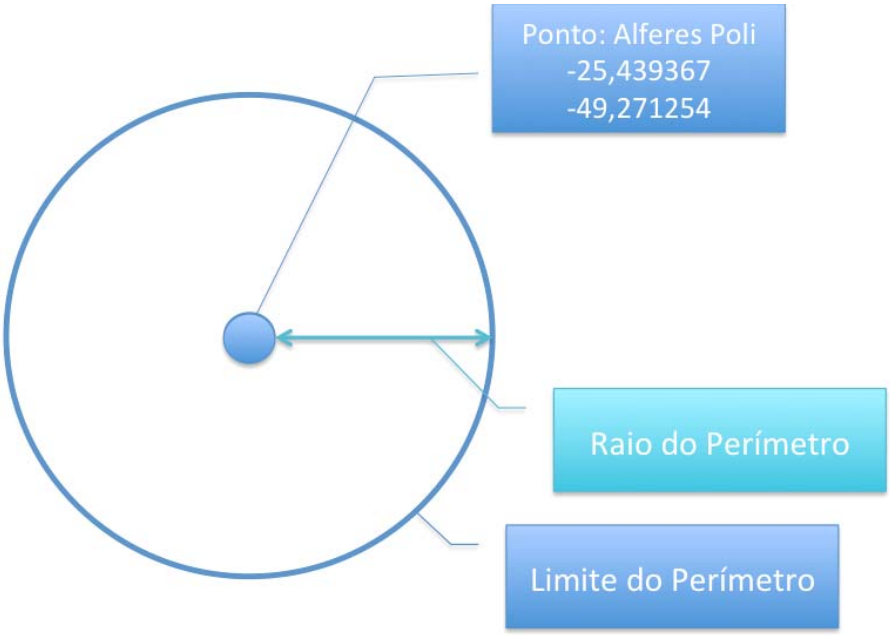

Figura 18 – Perímetro
Fonte: Autoria própria, 2015.

O perímetro satisfatório é uma circunferência de raio pré-definido, ao redor do ponto. Seu valor é inicializado com uma distância de 300 (trezentos) metros, podendo ser alterado pela Central a qualquer momento.

Para que não houvesse repetições de mensagens dentro do perímetro, foi necessário implementar dois estados para as mensagens por localização. Seus nomes, transições e ações são ilustrados pela figura abaixo.

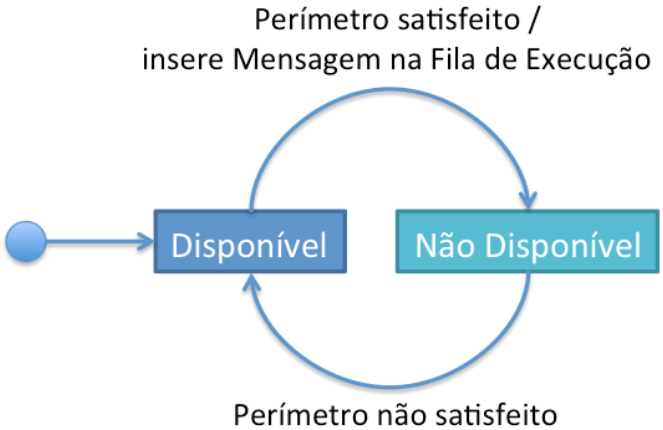

Figura 19 – Estados de Mensagem por Localização
Fonte: Autoria própria, 2015.

Mensagens em estado “Disponível” significa que estas estão disponíveis para serem agendadas e em seguida, executadas; todas as mensagens são inicializadas como tal. Mensagens em estado “Não Disponível” significa que não podem ser agendadas, sendo assim nunca serão executadas. Para que a mensagem saia do

estado “Disponível” para o “Não Disponível”, é necessário que as coordenadas da mensagem atinjam o limite ou estejam dentro do perímetro. Após a condição satisfeita, a mensagem altera seu estado e é inserida na fila de mensagens para síntese e execução automaticamente. Para que a mensagem volte a ser agendada, ela necessita retornar ao estado Disponível, onde a única maneira possível é através da detecção de um ponto fora do perímetro satisfatório.

As figuras a seguir apresentam a sequência de características e estados para uma aproximação e afastamento de um ônibus no ponto.

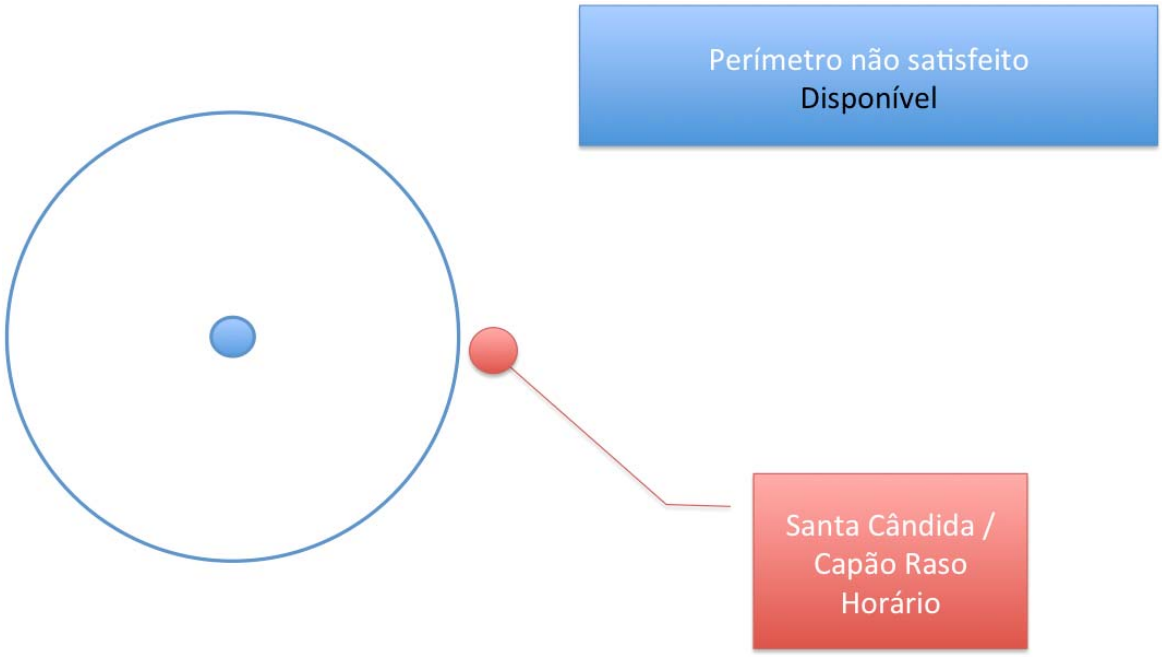

Figura 20 – Ônibus fora do perímetro da mensagem; Estado Disponível
Fonte: Autoria própria, 2015.

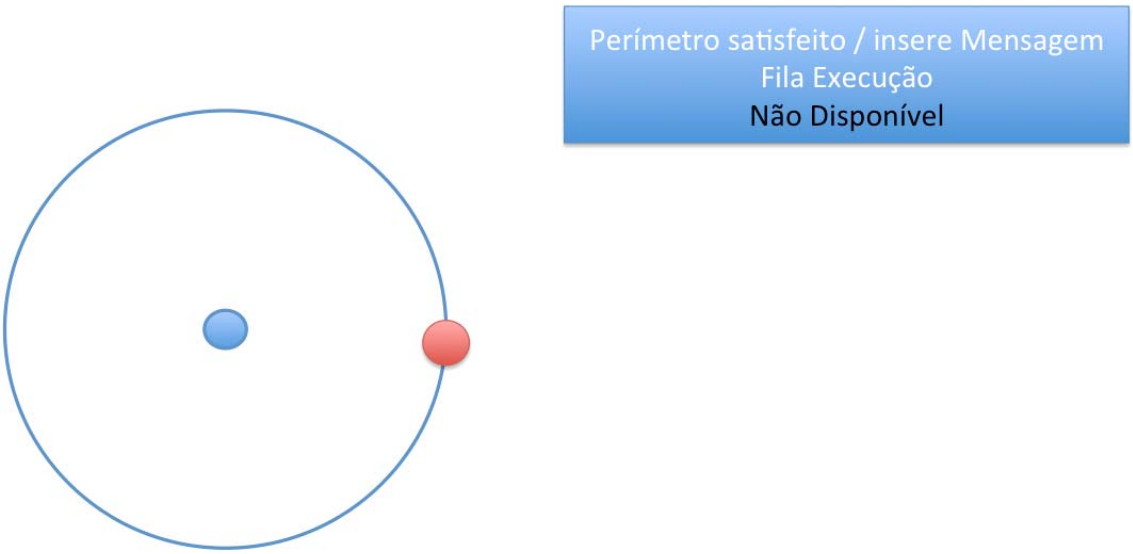

Figura 21 – Ônibus atinge o limite do perímetro; Estado Não Disponível
Fonte: Autoria própria, 2015.

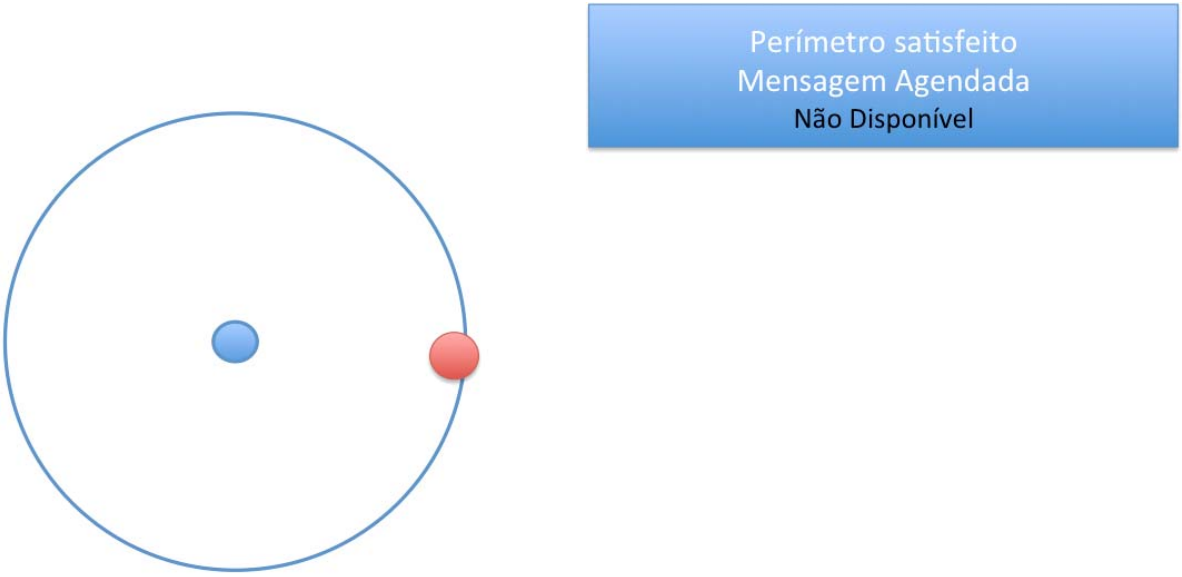

Figura 22 – Mensagem Agendada; Estado Não Disponível
Fonte: Autoria própria, 2015.

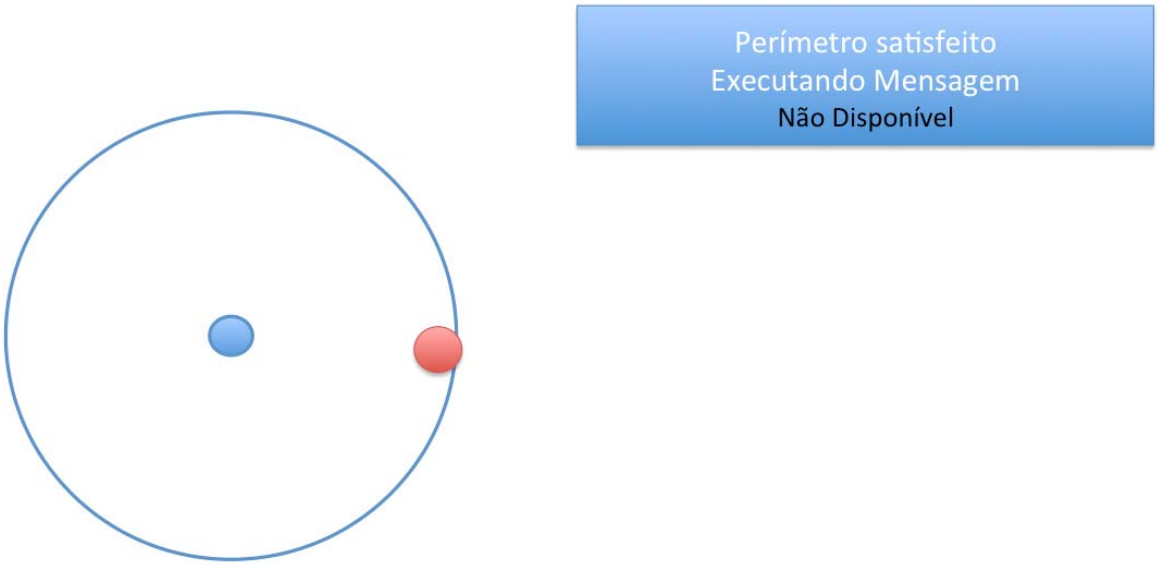

Figura 23 – Executando Mensagem; Estado Não Disponível
Fonte: Autoria própria, 2015.

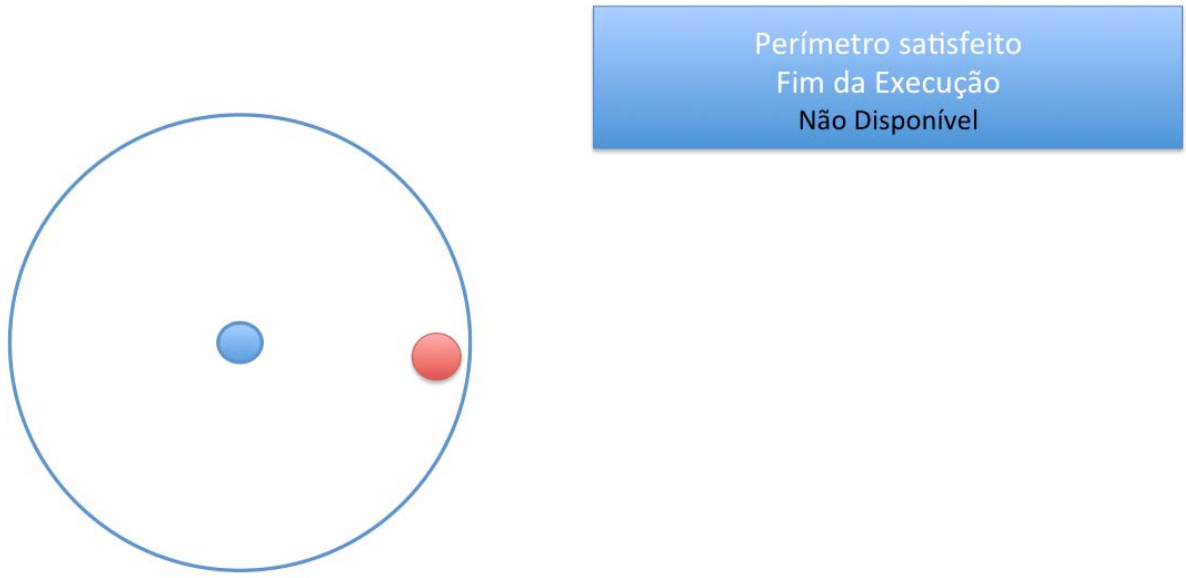

Figura 24 – Término da execução; Estado Não Disponível
Fonte: Autoria própria, 2015.

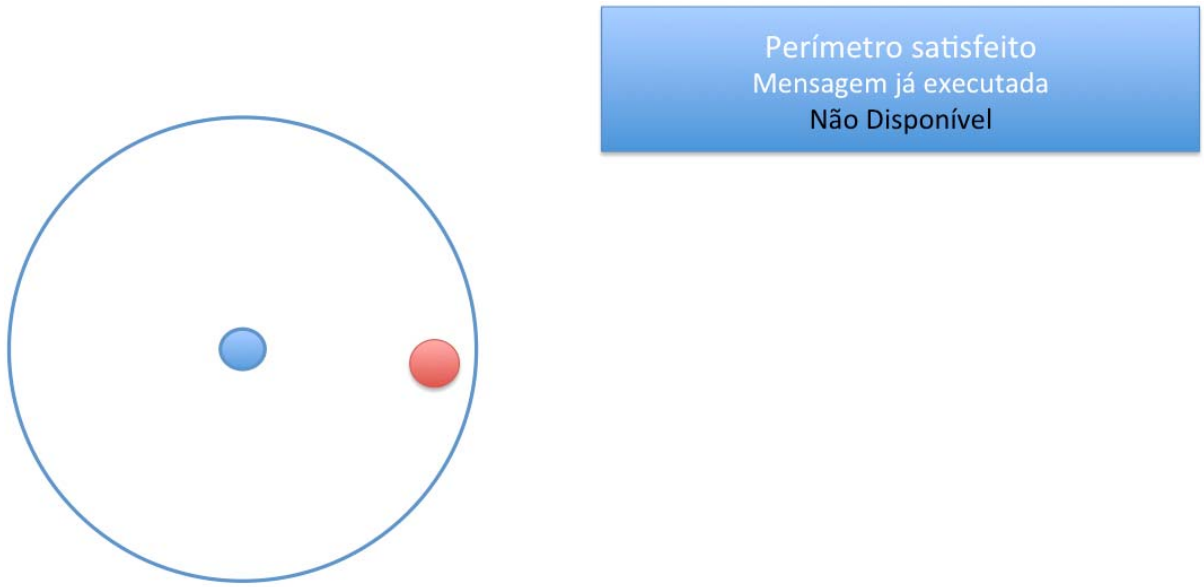

Figura 25 – Mensagem já executada; Estado Não Disponível
Fonte: Autoria própria, 2015.

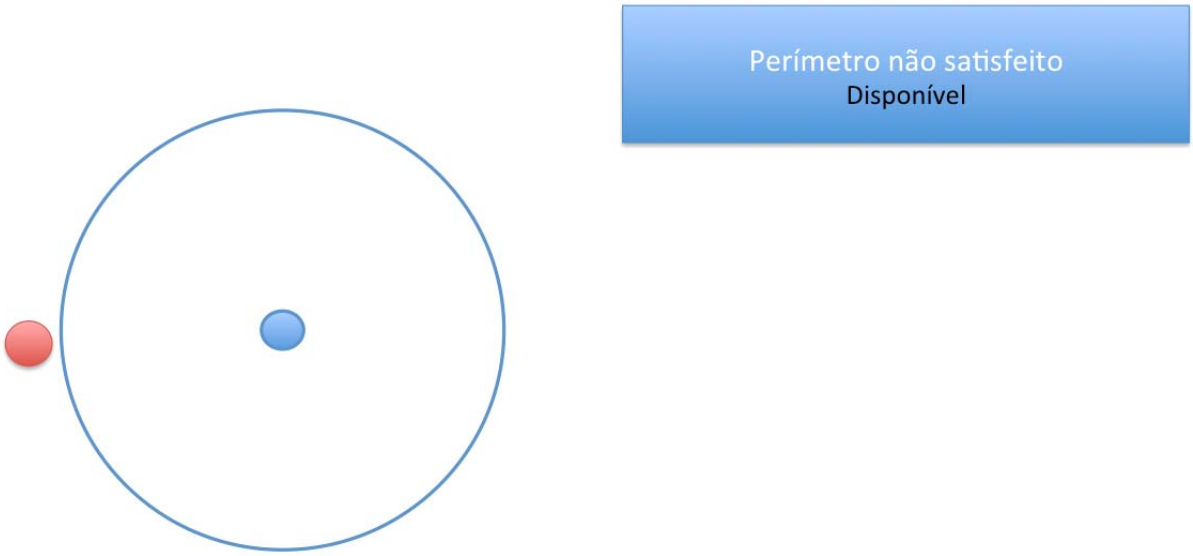

Figura 26 – Ônibus deixa o perímetro de satisfação; Estado Disponível
Fonte: Autoria própria, 2015.

É importante ressaltar que a relação entre a frequência de leitura e a velocidade do ônibus pode alterar o local de divulgação da mensagem, passando esta a ser executada em momento e local indevido. Acredita-se que uma relação ideal é quando a leitura do GPS é realizada a cada 500 (quinhentos) metros. Se o ônibus se locomove a 60 (sessenta) quilômetros por hora idealmente, o *polling* do GPS deve ser então realizado a cada 30 (trinta) segundos.

Para que as mensagens saiam do controle, o *polling* deve aumentar seu intervalo e/ou o ônibus sua velocidade. No caso inverso, onde o tempo do intervalo fosse reduzido e/ou o ônibus desacelerasse, as leituras seriam muito próximas podendo ocasionar coordenadas semelhantes. Sendo assim, é ideal que se mantenham as proporções entre a frequência e a velocidade: se a frequência abaixa, a velocidade deve abaixar também; se a frequência aumenta, a velocidade deve aumentar também.

5.1.2 Cliente/Servidor C++ (Processa)

O conceito cliente e servidor é bastante difundido entre os sistemas distribuídos (PASTOR, 2014), logo é possível encontrar facilmente diversas bibliotecas que implementem essa teoria.

No SiNDi foi utilizado protocolo TCP/IP para comunicação entre bibliotecas de Cliente e de Servidor. A porta utilizada foi 30000, onde a escolha desta foi aleatória para fins de validação da solução.

5.1.3 Festival C++ (Sintetiza)

Por se tratar de uma ferramenta de síntese de fala, ele carrega consigo a necessidade de inúmeras bibliotecas específicas, criadas exclusivamente para sua execução com diversas possibilidades de interface com outros sistemas. (BLACK; TAYLOR ;CALEY, 1999).

Para o SiNDi, foi decidido pela a utilização da API C/C++ onde ele estaria embarcado na aplicação. (item 28.4 da documentação do Festival (BLACK; TAYLOR ;CALEY, 1999)).

Apesar de fornecido pelos próprios desenvolvedores do Festival - documentação, bibliotecas, exemplos e orientações - a implementação não foi trivial.

Dentre os arquivos fornecidos alguns deles possuíam erros, o que impedia o teste de unidade do festival de ser realizado propriamente. Esse foi o motivo pela qual a unidade do Festival C++ foi a mais complexa, dentre todas as demais.

As principais funções oferecidas pela API são a função de inicialização e do sintetizador. A função de inicialização deve ser chamada antes de qualquer outra função para preparar o sintetizador. A função do sintetizador é responsável pela síntese do texto passado como parâmetro, retornando verdadeiro ou falso dependendo se a execução realizada corretamente.

Apesar de se ressaltar a importância da qualidade da fala, para a satisfação do usuário do transporte público de Curitiba (LANZONI, 2013), devido a limitação da ferramenta por não possuir síntese da língua Portuguesa, foi utilizado para a síntese padrões da língua Inglesa. Sendo assim, os arranjos ficaram distorcidos por sintetizar frases da língua Portuguesa com padrão Inglês.

Entretanto, esse item é propício para desenvolvimento em trabalhos futuros, já que o festival permite a criação de novas vozes independente de suas origens.

5.1.4 Estrutura de Dados

Algumas estruturas foram definidas para que o SiNDi possuísse um código otimizado seguindo os padrões de implementação de orientação objeto. Estas foram:

a. Estrutura de Coordenadas composta por dois números que identificam a

localização no globo;

b. Estrutura de Mensagens composta por:
    - identificador de mensagem;
        - código único para cada mensagem;
    - texto da mensagem;
        - mensagem definida pelo funcionário da URBS a ser falada nos ônibus;
    - tipo de mensagem;
        - definição se é uma mensagem educacional ou por localização;
    - coordenada GPS;
        - se mensagem por localização, esse campo indica o local onde a mensagem deve ser acionada;
    - se ela pode ser executada;
        - se mensagem por localização, ela só pode ser executada uma vez, quando dentro do perímetro aceitável para execução;
    - horário de última execução;
        - se mensagem educacional, esse campo indica a última vez que mensagem foi executada.
    - e se ela está na fila a para ser executada;
        - esse atributo é utilizado a fim de que a mesma mensagem não esteja duplicada na fila de execução;
c. Estrutura de elementos utilizados por todos os processos do SE, sendo estes:
    - Fila de mensagens;
        - armazena todos as mensagens recebidas da Central;
    - Fila de mensagens para execução;
        - armazena as mensagens temporariamente conforme o agendamento para serem executadas pelo festival; após execução as mensagens são removidas do vetor;
    - Fila de mensagens de emergência;
        - armazena as mensagens de emergência temporariamente para que sejam executadas; a mensagem é removida única e exclusivamente por algum pedido que venha da Central;

- Tempo de intervalo de mensagens educativas;
    - o define o valor em minutos do intervalo que as mensagens educativas devem ser executadas;
- Raio de limite do perímetro;
    - o define o valor em metros do perímetro aceitável para a execução das mensagens por localização;
- três semáforos para cada uma das filas;
    - o são implementados para que seja garantido a integridade na fila de mensagens em acessos concorrentes; cada variável é associada a um vetor diferente;

O diagrama de classes a seguir representa a relação entre os elementos apresentados nesse capítulo.

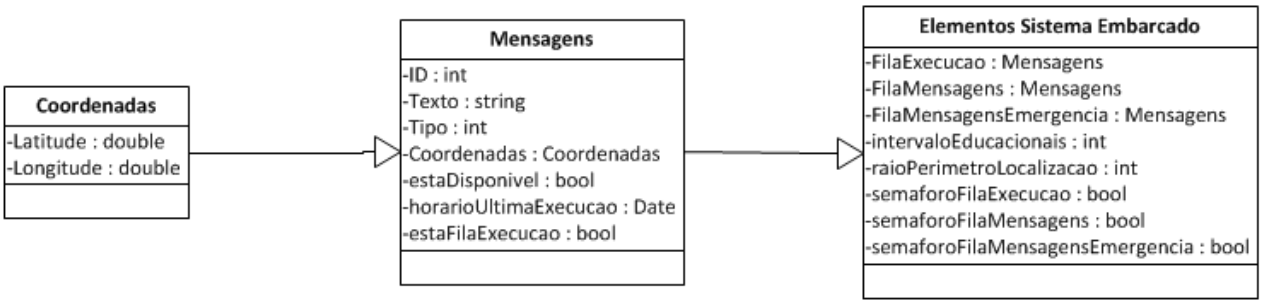

Figura 27 - Diagrama de Classes da estrutura de dados
Fonte: Autoria Própria, 2015.

As estruturas de dados -declarações ou agrupamentos de ações diretas implementadas pela linguagem de programação- não possuem a necessidade de teste de unidade. Neste caso, as verificações de erros são realizadas ao utiliza-las em elementos que implementam a lógica do programa.

5.1.5 Tratamento de Mensagem

A Central envia mensagem com a seguinte sintaxe:

tag messageID [latitude] [longitude] (messageText)

com itens entre colchetes sendo opcionais para mensagens educativas e de emergência.

Exemplos de mensagem educacional e de emergência:

```
MensagemEducacional 0 (Mensagem para ser Falada)

MensagemEmergencia 0 (Mensagem para ser Falada)
```

E exemplo de mensagem por localização:

```
MensagemLocalizacao 0 -25.4387 -49.2694 (Proxima parada, praca eufrasio correia)
```

O SE também implementa funções para tratamento do intervalo de repetição de mensagens educativas, limite do perímetro da localização e limpar lista de mensagens. A sintaxe para essas mensagens é:

tag [value]

com o item entre colchetes sendo opcional para os comando de limpar vetores.

Nesses casos as possibilidades de mensagens são exemplificadas a seguir:

```
intervaloEducacional 5
limiteLocalizacao 400
limparVetorExecucao
limparVetorEmergencia
limparVetorArmazenadas
```

O tratamento de mensagens recebidas é realizado na *thread* do servidor que, implementa o estado processa conforme especificado o item 4.3.1.

A sequência no processamento de mensagens é definida pelo fluxo abaixo.

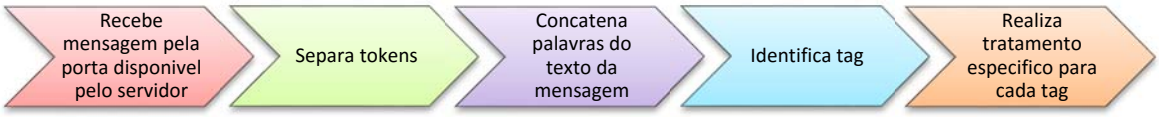

Figura 28 – Fluxo de tratamento de mensagens
Fonte: Autoria Própria, 2014

Os delimitadores de cada palavra são os espaçamentos, logo as mensagens de exemplo apresentadas acima retornaram um vetor de tamanho 6, 6 e 9, respectivamente. É importante ressaltar, que a quantidade de *tokens* da mensagem, depende da quantidade de palavras que existe no texto definido pela URBS.

Como não é interessante, em termos de organização de objetos, manter as palavras do texto da mensagem em *tokens* separados, o servidor faz um pós-tratamento procurando os parênteses dentre os tokens, para poder concatená-los novamente. Essa etapa não possui efetividade alguma quando nas mensagens recebidas, não houver parênteses.

5.2 Central

A Central é um elemento complementar no projeto, no qual seu objetivo é fornecer ao usuário interface com SE.

Seu desenvolvimento foi feito em C#, devido a disponibilidade da IDE, Visual Studio 2013 Ultimate, em integrar automaticamente o BD SQL Server 2012 definido no

Outra facilidade da ferramenta, é a programação por gráfico. Para gerar o layout ao usuário, o ambiente permite que o mesmo seja montado diretamente na janela, tornando intuitiva o desenvolvimento da interface da Central.

As linhas códigos necessárias na implementação foram para realizar comunicação TCP/IP com o SE e a conversão dos elementos de entrada na interface gráfica para elementos de tabelas no SQL.

Seu teste de unidade foi realizado em separado tanto para o BD quanto para a comunicação via *socket* do tipo TCP/IP.

6 RESULTADOS

Este capítulo apresenta os resultados dos testes descritos no item 4.1.6 que simulam o comportamento do usuário e exponhem as ações da Central e do SE. Neste contexto, apresentam-se 5 pontos reais para 2 linhas reais (Figura 30) para efeitos de teste.

A Figura 29, apresenta o SE com seus elementos, utilizados no decorrer do projeto.

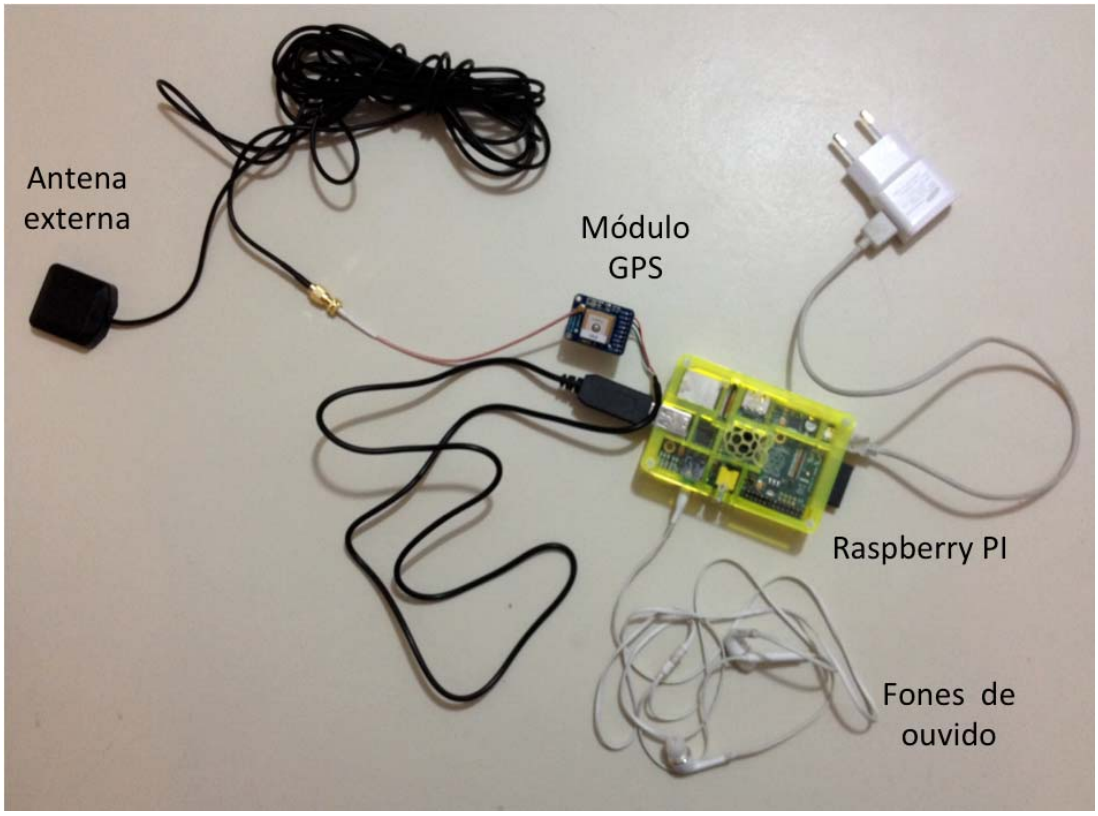

Figura 29 – Sistema Embarcado SiNDi
Fonte: Autoria Própria, 2014.

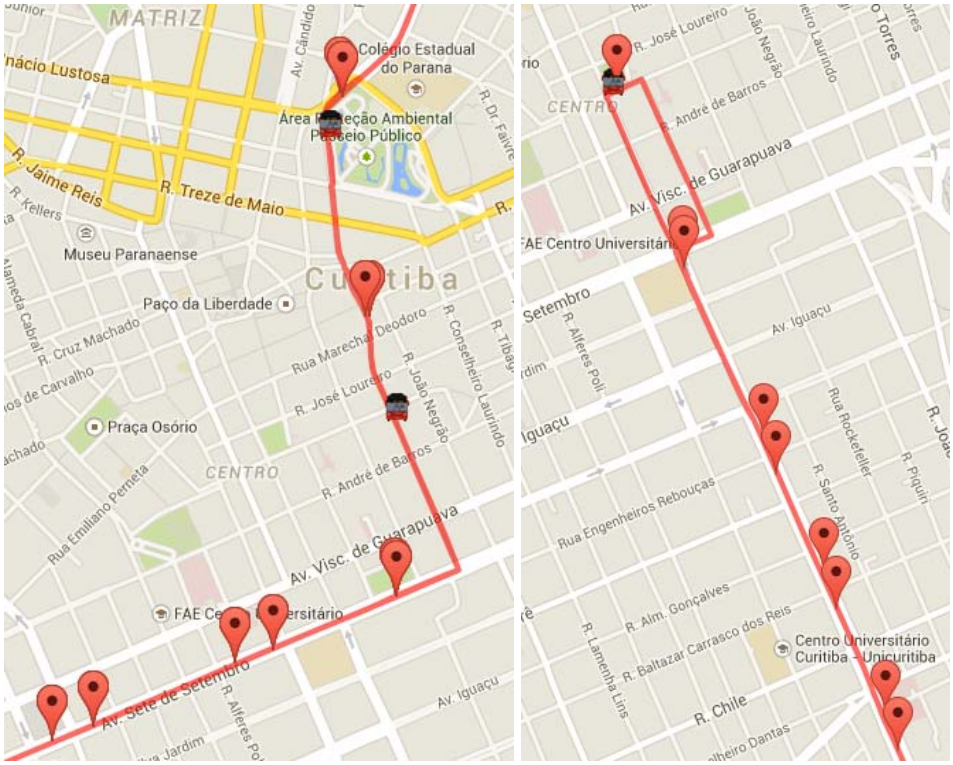

Figura 30 – Mapas das linhas 203 e 503
Fonte: Google Maps, 2014.

Esses itens que foram supostamente inseridos pelo usuário da URBS através do programa instalado na Central, os resultando no BD os dados como dispostos no Quadro 7, Quadro 8 e Quadro 9. Destaca-se que é previsto que, no momento da criação da Linha, o usuário já tenha cadastrado Pontos para poder relacionar com a Linha.

Quadro 7 – Dados para a tabela Ponto

nPontoID	cNomePonto	Latitude	Longitude
108022	Passeio Público	-25,42366	-49,26835
108024	Central	-25,429982	-49,267561
108025	Praça Eufrásio Correia	-25,437553	-49,266655
108028	Alferes Poli	-25,439367	-49,271254
108032	Praça Oswaldo Cruz	-25,44108	-49,275554
108065	Praça Carlos Gomes	-25,433706	-49,270224
108067	UTFPR	-25,439061	-49,268113
108070	Getúlio Vargas	-25,443629	-49,265721
108072	Almirante Gonçalves	-25,447435	-49,263822
108073	Conselheiro Dantas	-25,45149	-49,261923

Fonte: Autoria Própria, 2014.

Quadro 8 – Dados para tabela Linha

nLinhaID	cLinha
503	Boqueirão Horário
203	Santa Cândida / Capão Raso Horário

Fonte: Autoria Própria, 2014.

Quadro 9 – Dados para tabela LinhaPonto

nLinhaID	nPontoID	nSequenciaPonto
203	108022	1
203	108024	2
203	108025	3
203	108028	4
203	108032	5
503	108065	1
503	108067	2
503	108070	3
503	108072	4
503	108073	5

Fonte: Autoria Própria, 2014.

Do mesmo modo, o usuário deve cadastrar os ônibus, no qual ficariam dispostos no apresentado no Quadro 10. Destaca-se nesse caso, que é através do cadastro do ônibus que o usuário consegue informar qual a linha operante do mesmo, resultando com isso os itens dispostos no Quadro 11.

Quadro 10 – Dados para tabela Onibus

nOnibusID	cChassiOnibus	cIPOnibus	nPortaOnibus
1000	BWCA11J0Y4OOOOO1	192.168.1.100	30000
1001	BWCA11J0Y4OOOOO2	192.168.1.101	30000
1002	BWCA11J0Y4OOOOO3	192.168.1.102	30000
1003	BWCA11J0Y4OOOOO4	192.168.1.103	30000

Fonte: Autoria Própria, 2014.

Quadro 11 – Dados para tabela LinhaOnibus

nLinhaID	nOnibusID
203	1000
203	1001
503	1002
503	1003

Fonte: Autoria Própria, 2014.

Após a configuração dos pontos, linhas e ônibus, o usuário deve cadastrar as mensagens referentes aos mesmos. É importante ressaltar que, apesar de não ser mandatório, realizar as associações no momento do cadastro, assegura que o SE alocado para cada ônibus receberá instantaneamente a configuração realizada, caso o SE esteja em operação. Assim sendo, ao realizar o cadastro, o usuário possuirá no BD os dados alocados como no Quadro 12, Quadro 13 e Quadro 14.

Quadro 12 – Dados para tabela Mensagem

nMensagemID	cMensagem	bCurrent
2000	Próxima parada, estação Passeio Público. Desembarque por todas as portas.	1
2001	Próxima parada, estação Central Desembarque por todas as portas.	1
2002	Próxima parada, estação praça Eufrásio Correia Desembarque por todas as portas.	1
2003	Próxima parada, estação Alferes Poli. Desembarque pelas portas 2 e 4.	1
2004	Próxima parada, estação Praça Oswaldo Cruz. Desembarque pelas portas 2 e 4.	1
2005	Próxima parada, estação Praça Carlos Gomes. Desembarque por todas as portas.	1
2006	Próxima parada, estação UTFPR. Desembarque por todas as portas.	1
2007	Próxima parada, estação Getúlio Vargas. Desembarque pelas portas 2 e 4.	1
2008	Próxima parada, estação Almirante Gonçalves. Desembarque pelas portas 2 e 4.	1
2009	Próxima parada, estação Conselheiro Dantas. Desembarque pelas portas 2 e 4.	1
2010	Cuidado com furtos no interior do veículo.	1
2011	Esta linha estará inoperante no domingo, dia 14 de dezembro, das 6 às 10 horas da manhã, devido a Caminhada do Coração.	1
2012	Aguarde sempre o desembarque.	1

Fonte: Autoria Própria, 2014.

Quadro 13 – Dados para tabela MensagemPonto

nMessagemID	nPontoID
2000	108022
2001	108024
2002	108025
2003	108028
2004	108032
2005	108065
2006	108067
2007	108070
2008	108072
2009	108073

Fonte: Autoria Própria, 2014.

Quadro 14 – Dados para tabela MensagemLinha

nMensagemID	nLinhaID
2010	503
2011	503
2012	503
2010	203
2011	203
2012	203

Fonte: Autoria Própria, 2014.

Os pacotes a seguir serão enviados para os ônibus 1000 e 1001 com os IPs 192.168.1.100 e 192.168.1.101 e portas 30000 e 30001, respectivamente.

```
MensagemLocalizacao 2000 -25.42366 -49.26835 (Próxima parada, estação Passeio Público. Desembarque por todas as portas.)

MensagemLocalizacao 2001 -25.429982 -49.267561 (Próxima parada, estação Central Desembarque por todas as portas.)

MensagemLocalizacao 2002 -25.437553 -49.266655 (Próxima parada, estação praça Eufrásio Correia Desembarque por todas as portas.)

MensagemLocalizacao 2003 -25.439367 -49.271254 (Próxima parada, estação Alferes Poli. Desembarque pelas portas 2 e 4.)

MensagemLocalizacao 2004 -25.44108 -49.275554 (Próxima parada, estação Praça Oswaldo Cruz. Desembarque pelas portas 2 e 4.)
```

Os pacotes a seguir serão enviados para os ônibus 1002 e 1003 com os IPs 192.168.1.102 e 192.168.1.103 e portas 30002 e 30003, respectivamente.

```
MensagemLocalizacao 2005 -25.433706 -49.270224 (Próxima parada, estação Praça Carlos Gomes. Desembarque por todas as portas.)

MensagemLocalizacao 2006 -25.439061 -49.268113 (Próxima parada, estação UTFPR.  Desembarque por todas as portas.)

MensagemLocalizacao 2007 -25.443629 -49.265721 (Próxima parada, estação Getúlio Vargas. Desembarque pelas portas 2 e 4.)

MensagemLocalizacao 2008 -25.447435 -49.263822 (Próxima parada, estação Almirante Gonçalves. Desembarque pelas portas 2 e 4.)

MensagemLocalizacao 2009 -25.45149 -49.261923 (Próxima parada, estação Conselheiro Dantas. Desembarque pelas portas 2 e 4.)
```

As mensagens 2010, 2011 e 2012 são do tipo *Educacional*, e foram configuradas para serem executadas nas linhas 203 e 503. Assim sendo, todos os ônibus receberão os pacotes a seguir.

```
MensagemEducacional 2010 (Cuidado com furtos no interior do veículo.)

MensagemEducacional 2011 (Esta linha estará inoperante no domingo, dia 14 de dezembro, das 6
às 10 horas da manhã, devido a Caminhada do Coração.)

MensagemEducacional 2012 (Aguarde sempre o desembarque.)
```

Dentro da sequência apresentada, o programa se comportou de modo esperado, executando as mensagens de localização ao estarem 300 metros (valor padrão) dos pontos somente uma vez. Assim também como repetiam as mensagens educativas a cada 10 minutos (valor padrão), onde a cada 30 minutos todas as mensagens educativas se repetiam.

Para o próximo teste, o usuário reconfigurou o intervalo das mensagens educativas de 10 para 5 minutos e limite do perímetro do ponto das mensagens por localização de 300 para 100 metros.

Neste pedido os seguintes pacotes foram enviados a todos os ônibus.

```
intervaloEducacional 5
limiteLocalizacao 100
```

Com a mudança citada no parágrafo anterior, as mensagens de localização passaram a ser executadas bem próximo aos pontos cerca de 100 metros e mantinham a funcionalidade de ser executada apenas dentro do perímetro. Assim também as mensagens educativas alteraram seu comportamento passando a serem executadas cada 5 minutos, onde após 15 minutos todas as mensagens educativas se repetiam.

Outro teste realizado foi o carregamento de uma sequência de mensagens referente a uma linha, para um ônibus que operava em outra. Apresenta-se a situação onde, o ônibus 1000, que opera na linha 203, passará a operar na linha 503. Neste caso a Central coordenará uma sequência de ações para que a mudança seja realizada propriamente.

Para o usuário da URBS, a única ação necessária é a alteração do registro do ônibus pela interface da Central, enquanto no plano de fundo a seguinte sequência será executada:

1. Limpeza das listas de mensagens existentes no ônibus;

2. Alteração do BD na tabela LinhaOnibus;
3. Envio de pacotes configurando nova linha no ônibus.

Nos testes os resultados apresentados foram:

1. Limpeza das listas de mensagens existentes no ônibus;

Envio de pacotes de limpeza para o ônibus 1000;

```
limparVetorExecucao
limparVetorEmergencia
limparVetorArmazenadas
```

2. Alteração do BD na tabela LinhaOnibus;

Quadro 15 – Dados para tabela LinhaOnibus

nLinhaID	nOnibusID
503	1000
203	1001
503	1002
503	1003

Fonte: Autoria Própria, 2014.

3. Envio de pacotes configurando nova linha no ônibus.

Envio dos pacotes de mensagens a seguir, para os ônibus 1000 de IP 192.168.1.100 e portas 30000.

```
MensagemLocalizacao 2005 -25.433706 -49.270224 (Próxima parada, estação Praça Carlos Gomes. Desembarque por todas as portas.)

MensagemLocalizacao 2006 -25.439061 -49.268113 (Próxima parada, estação UTFPR.  Desembarque por todas as portas.)

MensagemLocalizacao 2007 -25.443629 -49.265721 (Próxima parada, estação Getúlio Vargas. Desembarque pelas portas 2 e 4.)

MensagemLocalizacao 2008 -25.447435 -49.263822 (Próxima parada, estação Almirante Gonçalves. Desembarque pelas portas 2 e 4.)

MensagemLocalizacao 2009 -25.45149 -49.261923 (Próxima parada, estação Conselheiro Dantas. Desembarque pelas portas 2 e 4.)
```

Acredita-se que os cenários testados satisfizem as expectativas levantadas junto à URBS ao apresentar um sistema flexível, com fácil configuração de mensagens e maior eficiência na comunicação com os usuários do transporte coletivo.

7 CONCLUSÃO

Neste trabalho de conclusão de curso utilizou-se de conhecimentos estudados no decorrer do curso de Engenharia de Computação: programação utilizando linguagens diversas: C++, C# e SQL; desenvolvimento e compilação de códigos em diferentes plataformas e ambientes: Linux, Windows, Eclipse, GCC e Visual Studio; teorias apresentadas em Sistemas Distribuídos: cliente/servidor; em Sistemas Embarcados: máquina de estados, *cross-compile* e interação com módulos periféricos; em Sistema Operacional: semáforos; em Engenharia de Software: diagramas UML; e em Estrutura de Dados: programação orientada objeto. Também apropiou-se de temas e ferramentas não abordados no curso como, síntese de fala.

O SiNDi é projeto composto de um SE (Sistema Embarcado) que deve ser instalado no ônibus, e uma Central que se comunica com o SE utilizada por funcionários da URBS para configuração de novas mensagens.

O cliente – Central – é um computador onde se implementou funções de cadastro de ônibus, linhas, pontos e mensagens. Também pela interface, é possível configurar as mensagens de acordo com seu objetivo, podendo esta ser uma mensagem temporizada ou dependente da localização. Outrora, essas características são assimiladas de acordo com a relação explicita da mensagem com ônibus, linha ou ponto.

O servidor – SE, por sua vez, é composto por um hardware e software. O hardware é o conjunto dos elementos Raspberry PI Model B, módulo *wi-fi*, módulo GPS, antena externa para GPS e alto-falantes. O software é responsável por receber os pacotes enviados pelo cliente, escalona-los e sintetizar as mensagens conforme sua configuração. Para a síntese foi utilizada a biblioteca do Festival, permitindo que a síntese fosse realizada diretamente pelo software.

Na arquitetura proposta, a comunicação Central-SE desempenha um papel secundário, pois havendo mensagens configuradas, o SE tem um comportamento autônomo. Porém, sem esta comunicação, a flexibilidade do sistema é reduzida, ficando a mercê do seu reestabelecimento.

O protocolo TCP/IP funcionou adequadamente e foi testada com um módulo *wi-fi*, comprovando conceitualmente que é possível a conexão remota da Central com o SE.

Socialmente, este trabalho espera contribuir para uma melhor aceitação do transporte coletivo ao promover melhor comunicação com os usuários.

Do ponto de vista tecnológico, o projeto inova com a implementação de um processo moderno envolvendo novas tecnologias no setor de transporte público. Do econômico, o SiNDi reduz o tempo no qual a URBS demorava para realizar alterações de mensagens nos ônibus de 10 dias, para alguns minutos, além da redução de gastos com impressão e distribuição de avisos para a notificação dos usuários da RIT.

Por fim, acredita-se que esta monografia tenha atingido seu objetivo proposto ao entregar um sistema:

- flexível, permitindo configuração e reconfiguração de mensagens de maneira instantânea pela URBS;
- com acesso remoto, devido a implementação do protocolo TCP/IP;
- que utiliza um receptor GPS para identificar os locais onde mensagens devem ser reproduzidas;
- e que utiliza de um software de síntese de fala para gerar áudio-mensagens nos ônibus da RIT.

7.1 Trabalhos Futuros

Essa monografia abre possibilidade de trabalhos futuros nos seguintes aspectos:

- Participação dos usuários para avaliação dos componentes informativos;
- Implementação de nova voz em Português, melhorando a qualidade da fala;
- Implementação do sistema para pontos de ônibus e estações-tubos;
- Substituição de rede internet por rede ad-hoc entre todos os elementos da RIT;
- Implementação do conceito apresentado no SiNDi para outros modais

de transporte;

- Estudo de viabilidade da implementação real do SiNDi em toda RIT.
- Implementação de alertas que notifiquem a URBS sobre possíveis falhas no sistema: falha de comunicação, recepção de satélite GPS perdida, mensagem não configurada, falha do SE e etc.

8 REFERÊNCIAS BIBLIOGRÁFICAS

ALBUQUERQUE, Almir dos S. **Análise Comparativa dos Métodos de Sintetização de Voz**. 2001. 76 f. Dissertação (Mestrado em Ciência da Computação) - Área de Concentração de Sistemas de Computação, Universidade Federal de Santa Catarina, Florianópolis, 2001. Disponível em: < http://repositorio.ufsc.br/bitstream/handle/123456789/80028/206429.pdf?sequence=1&isAllowed=y>. Acesso em: 30 mar. 2015.

ANTONELLI, Diego; BARÃO, Gisele. Sistema de ônibus curitibano perdeu 14 milhões de usuários em 4 anos. **Gazeta do Povo**, Curitiba, 2 jun. 2012. Disponível em: <http://www.gazetadopovo.com.br/vida-e-cidadania/sistema-de-onibus-curitibano-perdeu-14-milhoes-de-usuarios-em-4-anos-1ug8szlhgxi2gjunpt88pbqdq>. Acesso em: 15 abr. 2013.

BLACK, Alan W.; LENZO, Kevin A. Flite: A Small Fast Run-Time Synthesis Engine. In: ISCA TUTORIAL AND RESEARCH WORKSHOP ON SPEECH SYNTHESIS, PERTHSHIRE. 2001. p. 20-4.

BLACK, Alan W.; TAYLOR, Paul; CALEY, Richard. **The Festival Speech Synthesis System: System documentation**. 1999. Edition 1.4, for Festival Version 1.4.0. 17th June 1999. Disponível em: <http://www.cstr.ed.ac.uk/projects/festival/manual/>. Acesso em: 13 dez. 2014.

BRAMBILLA, Flávio R.; PEREIRA, Luciana V.; PEREIRA, Paula B. Marketing de Relacionamento: Definição e Aplicações. **INGEPRO – Inovação, Gestão e Produção**, v. 2, n. 12, dez. 2010, ISSN 1984-6193. Disponível em: <http://www.ingepro.com.br/Publ_2010/Dez/306-941-1-PB.pdf>. Acesso em: 13 dez. 2014.

CUERVO, Luciane. **Introdução à Fisiologia da Voz**. Departamento de Música – UFRGS. Práticas Vocais para a Educação Musical. Porto Alegre. 2010. Material produzido para fins didáticos. Disponível em: <http://www.ufrgs.br/musicalidade/midiateca/praticas-musicais-vocais-e-instrumentais/praticas-vocais/ovas-praticas-vocais/fisiologia>. Acesso em: 23 mar. 2015.

DUARTE, Fábio; LIBARDI, Rafaela; SÁNCHEZ, Karina. **Introdução à mobilidade urbana**. Curitiba: Juruá, 2007.

ECLIPSE. **Conceito: Modelo de Caso de Uso**. Disponível em: <http://epf.eclipse.org/wikis/openuppt/openup_basic/guidances/concepts/use_case_model,_2jyfUAhVEduRe8TeoBmuGg.html>. Acesso em: 27 abr. 2015.

EMBARQ BRASIL. **De cá para lá: um guia criativo de marketing BRT para atrair e cativar usuário**. 2011. Disponível em: <http://www.ctsbrasil.org/marketingbrt>. Acesso em: 15 abr. 2013.

FALBO, Ricardo de A. **Engenharia de Software.** 2005. 146 f. Notas de Aula - Universidade Federal do Espírito Santo. Disponível em :

<http://www.inf.ufes.br/~falbo/download/aulas/es-g/2005-2/NotasDeAula.pdf>. Acesso em: 13 dez. 2014.

FANT, Gunnar. **Acoustic Theory of Speech Production**. Mounton: The Hague, 1960.

FOLDOC. **Spiral Model.** [1997?]. Disponível em: <http://foldoc.org/spiral+model>. Acesso em: 13 dez. 2014.

GABRIEL, Philipe. Como funciona o GPS?. **Oficina da Net**, 12 abr. 2014 - 10h30. Disponível em: <http://www.oficinadanet.com.br/post/12406-como-funciona-o-gps>. Acesso em: 13 dez. 2014.

GARMIN. **What is GPS?**. Disponível em: <http://www8.garmin.com/aboutGPS/>. Acesso em: 14 abr. 2015.

GPS.Gov. **GPS Accuracy**. Disponível em: <http://www.gps.gov/systems/gps/performance/accuracy/>. Acesso em: 14 abr. 2015.

GUSMÃO, Bruna. A maneira mais eficaz de melhorar a pronúncia em inglês. **Blog Bruna Gusmão**, 2013. Disponível em : < http://blogs.odiario.com/brunagusmao/2013/07/23/a-melhor-maneira-de-melhorar-a-pronuncia-em-ingles/>. Acesso em: 9 abr. 2015.

HENTZ, Augusto H.. **Compressão de Bancos de Fala para Sistemas de Síntese Concatenativa de Alta Qualidade**. 2009. 63 f. Dissertação (Mestre em Engenharia Elétrica) - Área de Concentração em Comunicações e Processamento de Sinais. Universidade Federal de Santa Catarina, Florianópolis, 2009. Disponível em : <https://repositorio.ufsc.br/bitstream/handle/123456789/92460/275012.pdf?sequencs=1>. Acesso em: 7 abr. 2015.

INSTITUTO DE PESQUISA ECONÔMICA APLICADA. **Sistema de indicadores de percepção social (SIPS)**: Mobilidade urbana. Brasília, 2011. Disponível em: <http://www.ipea.gov.br>. Acesso em: 15 abr. 2014.

INTERNATIONAL ASSOCIATION OF PUBLIC TRANSPORT (UITP). Passenger Information. **Core Brief** , Março 2001a. Disponível em: <http://www.uitp.org>. Acesso em: 15 abr. 2013.

_____. Access to Public Transport. **FOCUS** – A UITP Position Paper, Junho 2001b. Disponível em: <http://www.uitp.org>. Acesso em: 15 abr. 2014.

_____.Marketing as an investment in greater client satisfaction and better benefits. **FOCUS** – A UITP Position Paper, Setembro 2002. Disponível em: <http://www.uitp.org>. Acesso em: 15 abr. 2014.

LANZONI, Cristine de O. **Gestão do design no transporte público de Curitiba**: Um estudo de caso do desenvolvimento do sistema de informação ao usuário das paradas de ônibus tipo abrigo. 2013. 185 f. Dissertação (Mestrado em Design) -

Setor de Artes, Comunicação e Design, Universidade Federal Do Paraná, Curitiba, 2013. Disponível em : <http://dspace.c3sl.ufpr.br/dspace/bitstream/handle/1884/33340/R%20-%20D%20-%20CRISTINE%20DE%20OLIVEIRA%20LANZONI.pdf?sequence=1>. Acesso em: 30 nov. 2014.

LEMOS, Mariana. Como funciona o aparelho fonador?. **Galileu**, 2014. Disponível em: <http://revistagalileu.globo.com/Revista/Galileu/0,,EDR82619-7946,00.html>. Acesso em 30 de março de 2015.

LIMA, Mariana A. De C. S. Laringe. **Mundo Educação**. Disponível em: <http://www.mundoeducacao.com/biologia/laringe.htm>. Acesso em: 27 mar. 2015.

MARTINS, Elaine. Como funciona o GPS?. **TEC MUNDO**, 10 ago. 2009 - 22h25. Disponível em: <http://www.tecmundo.com.br/gps/2562-como-funciona-o-gps-.htm>. Acesso em: 13 dez. 2014.

MICROSOFT. **Microsoft Speech Plataform**. Disponível em: <https://msdn.microsoft.com/en-us/library/office/hh361572(v=office.14).aspx>. Acesso em: 14 abr. 2015.

MORI, Kentaro. Waseda Talker: um robô com cordas vocais. **Science Blogs**, 2 out. 2008. Disponível em: <http://scienceblogs.com.br/100nexos/2008/10/waseda-talker-um-rob-com-cordas-vocais/>. Acesso em: 8 abr. 2015.

NAGLE, Edson J.; CHIQUITO, José G. Síntese de Sinais de Fala Usando o Sintetizador de Formantes de Klatt - Laboratório de Processamento de Sinais – DECOM-FEE-UNICAMP, 1993. Campinas. **Anais**..., 1993. 718 p.. Disponível em : < http://www.eletrica.ufpr.br/anais/sbrt/SBrT11/SBrT_1993v2_048.pdf>. Acesso em: 8 abr. 2015.

NASA Star Child. **Stephen Hawking Portrait.** [19--?] Disponível em: <http://starchild.gsfc.nasa.gov/Images/StarChild/scientists/hawking.jpg>. Acesso em: 13 dez. 2014.

ORACLE. **CRM – Customer Relationship Management.** Disponível em : <https://www.oracle.com/applications/customer-experience/crm/index.html>. Acesso em: 13 dez. 2014.

ORSINI, Lauren. Arduino Vs. Raspberry Pi: Which Is The Right DIY Platform For You? **ReadWrite**, 4 mai. 2014. Disponível em: <http://readwrite.com/2014/05/07/arduino-vs-raspberry-pi-projects-diy-platform>. Acesso em: 15 abr. 2015.

PASTOR, Luis P. R. Sistemas Distribuídos: Conceitos fundamentais e técnicas para implementações em Java RMI. **Revista e-f@tec**, vol. 04, nº 01, 2014, ISSN 2317-451X. Disponível em : <http://www.fatecgarca.edu.br/revista/volume4/artigos/4.pdf>. Acesso em: 13 dez. 2014.

PETTERSSON, Rune. **It depends: ID-Principles and Guidelines**. 2 ed. Tullinge: Institute for Infology, 2007.

PREFEITURA MUNICIPAL DE CURITIBA. **Urbanização de Curitiba, S/A**. 2014. Disponível em: <http://goo.gl/zIymoj>. Acesso em: 20 jul. 2014.

ROCHA, Camila. **Estudo da qualidade de software na Metodologia V-model e sua interação com metodologias ágeis (SCRUM)**. 2011. Faculdade De Tecnologia de São Paulo. São Paulo. 2011. Disponível em : <http://www.fatecsp.br/dti/tcc/tcc0028.pdf>. Acesso em: 13 dez. 2014.

ROYCE, Winston W. **Managing the development of large software systems**. 1970. The Institute of Electrical and Electronics Engineers. Disponível em: <http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf>. Acesso em: 13 dez. 2014.

SILVA, Bruno C. R.; CÂNDIDO, Luciano A. A. **Sistema de Controle Residencial Baseado na Plataforma Arduino.** 2011. 49 f. Monografia (Ciência da Computação) , Instituto Unificado de Ensino Superior Objetivo, Goiânia. 2011. Disponível em: <https://pt.scribd.com/doc/80802432/5/Eletronica-e-Computacao>. Acesso em: 15 abr. 2015.

SIMÕES, Flávio O. **Implementação de um Sistema de Conversão Texto-Fala para o Português do Brasil***.* 1999. 186 f. Dissertação (Mestre em Engenharia Elétrica), Universidade Estadual de Campinas, Campinas. 1999. Disponível em : <http://www.decom.fee.unicamp.br/lpdf/teses_pdf/Tese-Mestrado-Flavio_Olmos_Simoes.pdf>. Acesso em: 8 abr. 2015.

SourceForge.NET. **eSpeak text to speech**. [20--?]. Disponível em: <http://espeak.sourceforge.net/>. Acesso em: 9 abr. 2015.

TAKANISHI LAB. **Anthropomorphic Talking Robot:** Waseda Talker Series. Disponível em: <http://www.takanishi.mech.waseda.ac.jp/top/research/voice/index.htm>. Acesso em: 8 abr. 2015.

TAYLOR, Paul; BLACK, Alan W.; CALEY, Richard. **The Architecture of the Festival Speech Synthesis System**. In: 3rd ESCA Workshop In Speech Synthesis. 1998. p.147-151.

TRUYTS et al. Steps for requirements writing. **Product: Management & Development - PMD**, vol. 10, nº 02, 2012, ISSN 1676-4056. Disponível em : <http://pmd.hostcentral.com.br/revistas/vol_10/nr_2/v10n2a05.pdf>. Acesso em: 13 dez. 2014.

URBS. **URBS em números.** Disponível em: <http://www.urbs.curitiba.pr.gov.br/institucional/urbs-em-numeros>. Acesso em: 29 nov. 2014.

ZUFFO, Felipe; PISTORI, Hemerson. **Tecnologia Adaptativa e Síntese de Voz: Primeiros Experimentos**. 2004. Disponível em: <http://www.pcs.usp.br/~lta/artigos/zuffo_wsl2004.pdf>. Acesso em: 13 dez. 2014.