

MEC-SETEC
INSTITUTO FEDERAL DE MINAS GERAIS – CAMPUS FORMIGA
Curso de Ciência da Computação

Um Protótipo para Disseminação de Eventos Rodoviários

Sávio Cardoso Caetano

Orientador: Prof. Me. Everthon Valadão

Formiga - MG

2017

Sávio Cardoso Caetano

UM PROTÓTIPO PARA DISSEMINAÇÃO DE EVENTOS RODOVIÁRIOS

Monografia apresentada ao Curso de Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais - Campus Formiga, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Área de concentração: Computação.

Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais - Campus Formiga
Ciência da Computação

Orientador: Prof. Me. Everthon Valadão

Formiga - MG

2017

004 Caetano, Sávio Cardoso.
Um protótipo para disseminação de eventos rodoviários /Sávio
Cardoso Caetano. -- Formiga : IFMG, 2017.
75p. : il.

Orientador: Prof. Me. Everthon Valadão
Trabalho de Conclusão de Curso – Instituto Federal de Educação,
Ciência e Tecnologia de Minas Gerais – *Campus* Formiga.

1. Computação. 2. Disseminação epidêmica, 3. VANET. 4. V2V. 5.
Eventos rodoviários. I. Título.

CDD 004

SÁVIO CARDOSO CAETANO

**UM PROTÓTIPO PARA DISSEMINAÇÃO DE EVENTOS
RODOVIÁRIOS**

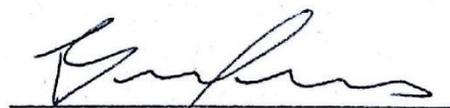
Trabalho de Conclusão de Curso apresentado ao
Instituto Federal de Minas Gerais-Campus Formiga,
como Requisito parcial para obtenção do título de
Bacharel em Ciência da Computação.

Aprovado em: 09 de junho de 2017.

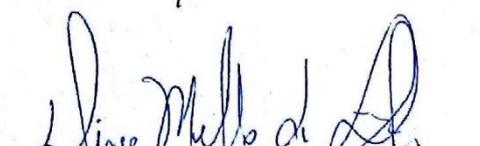
BANCA EXAMINADORA



Prof.º Everthon Valadão dos Santos



Prof.º Bruno Ferreira



Prof.º Diego Mello da Silva

Dedico este trabalho aos meus pais Sérgio Luís e Carla Cardoso, que sempre lutaram pela minha educação e tornaram possível que eu chegasse até aqui. E também à todas as pessoas que contribuíram de alguma forma. Sem eles essa conquista não seria possível.

Agradecimentos

Agradeço primeiramente a Deus pelo dom da vida e por estar sempre olhando por mim.

Agradeço aos meus pais Sérgio Luís e Carla Cardoso, por todo apoio oferecido, não só durante a realização deste trabalho, como durante a vida inteira.

Agradeço ao meu orientador Everthon Valadão, pelas dicas, conselhos e por estar sempre pronto para ajudar durante todo o desenvolvimento do projeto.

Agradeço a todos os amigos pelos conselhos recebidos, em especial aos amigos Alex, Eduardo e Rodrigo que dedicaram algumas horas para me ajudar com as dificuldades encontradas e/ou me ajudaram compartilhando conhecimento.

Agradeço a Ana Gabriela por todo companheirismo e paciência durante esse período de desenvolvimento e também por sempre ter me incentivado, acreditando que essa conquista seria possível.

Resumo

Infelizmente, ainda é grande o número de acidentes que ocorrem em rodovias, fazendo vítimas e tirando vidas. Para fomentar uma maior segurança no trânsito, podem ser adotadas VANETs (Vehicular Ad Hoc Network), que são redes de comunicação sem fio de propósito específico: prover troca de informações entre veículos. As VANETs têm seus nós compostos por veículos e seus equipamentos de comunicação móvel e por equipamentos fixos que podem estar presentes ao longo das vias. Este trabalho apresenta um protótipo para disseminação de eventos rodoviários, através da comunicação sem fio entre veículos (V2V – Vehicle-to-Vehicle) feita de forma epidêmica, propagando os eventos por todos os nós da rede de forma rápida e escalável. Para tal, foi desenvolvido um app (aplicação mobile) para enviar e exibir tais alertas ao motorista, utilizando como interface um dispositivo móvel com sistema operacional Android. O app interage com o firmware do protótipo, realizado em microcontrolador ESP-8266 na plataforma nodeMCU, que provê comunicação na faixa de 2,4 GHz. Foram considerados requisitos de qualidade de serviço para aplicações de segurança no trânsito e desenvolvidos algoritmos para a disseminação epidêmica dos eventos rodoviários na rede. Utilizando equipamentos de custo acessível e software livre, foi possível construir um sistema capaz de disseminar eventos rodoviários.

Palavras-chave: Disseminação epidêmica. VANET. V2V. Eventos rodoviários. Dispositivo móvel Android.

Abstract

Unfortunately, there are still a lot of accidents that occur on highways, resulting in injuries and deaths. To promote greater traffic safety, VANETs (Vehicular Ad Hoc Networks), which are wireless communication networks with specific purposes, can be used to provide information exchange between vehicles. The VANETs nodes are composed by vehicles and their mobile communication equipment and by fixed equipment that may be present along the roads. This paper presents a prototype for the dissemination of road events, through the wireless communication between vehicles (V2V - Vehicle-to-Vehicle) made in an epidemic way, propagating events across all nodes of the network in a fast and scalable manner. To do so, an app (mobile application) was developed to send and display such alerts to the driver, using a mobile device with Android operating system as its interface. The app interacts with the firmware of the prototype, performed by a microcontroller ESP-8266 in platform nodeMCU, that provides communication in the range of 2.4 GHz. Quality of service requirements were considered for traffic safety applications and algorithms were developed for the epidemic dissemination of road events in the network. Using affordable equipment and free software, it was possible to build a system capable of disseminating road events.

Keywords: Epidemic dissemination. VANET. V2V. Road events. Android mobile device.

Lista de abreviaturas e siglas

2G	Segunda geração de tecnologia móvel celular;
3G	Terceira geração de tecnologia móvel celular;
4G	Quarta geração de tecnologia móvel celular;
AFH	Adaptive Frequency Hopping;
AP	Access Point;
APP	Aplicação Mobile;
BLE	Bluetooth Low Energy;
DENATRAN	Departamento Nacional de Trânsito;
DSRC	Dedicated Short Range Communications;
DTN	Delay Tolerant Networks (Redes Tolerante a Interrupção);
FCC	Federal Communications Commission;
GPS	Global Positioning System (Sistema de Posicionamento Global);
HCI	Host Controller Interface;
I/O	Input / Output;
IoT	Internet of Things (Internet das coisas);
IPEA	Instituto de Pesquisa Econômica Aplicada;
ISM	Industrial Scientific Medical;
LNA	Low noise amplifier;
MAC	Medium Access Control;
MANET	Mobile Ad Hoc Network;
OBD-II	On Board Diagnostic-II;
OBU	On Board Unit;
OMS	Organização Mundial de Saúde;

RSU	Road Side Unit;
SIG	Special Interest Group;
STA	Station Ad Hoc Mode;
TCC	Trabalho de conclusão de curso;
TTL	Transistor-Transistor Lógic;
UART	Universal asynchronous receiver/transmitter;
UHF	Ultra High Frequency;
UTC	Universal Time Coordinated;
V2V	Vehicle-to-Vehicle;
V2I	Vehicle-to-Infrastructure;
V2X	Vehicle-to-Everything (V2V + V2I)
VANET	Vehicular Ad Hoc Network;
WAVE	Wireless Access in the Vehicular Environment
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network

Sumário

1	INTRODUÇÃO	17
1.1	Motivação	18
1.1.1	Solução Proposta	19
1.1.2	Trabalhos Relacionados	19
1.2	Objetivos	21
1.2.1	Objetivo Geral	21
1.2.2	Objetivos Específicos	22
1.3	Organização do Texto	22
2	FUNDAMENTAÇÃO TEÓRICA	23
2.1	Redes móveis sem fio	23
2.1.1	<i>Vehicle Ad-hoc Network (VANET)</i>	23
2.1.2	Arquiteturas para VANETs	24
2.1.2.1	Arquitetura V2V	24
2.1.2.2	Arquitetura V2I	24
2.1.2.3	Arquitetura V2X	24
2.1.2.4	Arquitetura WAVE	25
2.1.3	Aplicações de VANETs	27
2.1.3.1	Segurança no Trânsito	28
2.1.3.2	Aplicações de Entretenimento	28
2.1.3.3	Assistência ao Motorista	29
2.2	Tecnologias de Comunicação sem Fio	29
2.2.1	Wi-Fi	29
2.2.2	Bluetooth	30
2.2.2.1	Exemplos de uso de Bluetooth em VANETs	32
2.3	Tecnologias de Posicionamento	32
2.3.1	<i>Global Positioning System (GPS)</i>	33
2.3.2	<i>Assisted GPS (AGPS)</i>	33
2.4	Disseminação de Dados	33
2.5	Eventos Rodoviários	35
2.6	Desenvolvimento Móvel	35
2.6.1	O Ecosistema Android	36
3	METODOLOGIA	37
3.1	Revisão Bibliográfica	37

3.2	Métodos	37
3.2.1	Definição dos eventos rodoviários a serem suportados	37
3.2.2	Encapsulamento do evento em pacote com formato padronizado	38
3.2.3	Definição do kit para desenvolvimento do protótipo da OBU	39
3.2.3.1	DevKit Arduino Pro Mini e módulo nRF24L01+	39
3.2.3.2	<i>DevKit nodeMCU</i> com microcontrolador ESP8266-12e	39
3.2.4	Definição da <i>suíte</i> de desenvolvimento do <i>app mobile</i> : Android Studio	40
3.2.5	Protótipo para exibição dos alertas no <i>app</i>	40
3.2.6	Desenvolvimento do <i>firmware</i> para comunicação <i>app</i> -OBU	41
3.2.7	Desenvolvimento do <i>firmware</i> para comunicação entre OBUs	42
3.2.8	Desenvolvimento do protocolo para disseminação epidêmica dos eventos	42
3.3	Materiais	43
3.3.1	Kit de desenvolvimento Arduino	43
3.3.1.1	Arduino Mega 2560	43
3.3.1.2	Arduino Pro Mini	43
3.3.2	Módulo <i>wireless</i> 2.4 GHz (nRF24L01+)	45
3.3.3	Módulo Bluetooth 2.1 (HC-06)	46
3.3.4	Kit de desenvolvimento NodeMCU 1.0	46
4	PROJETO	49
4.1	Escopo	49
4.2	Requisitos	49
4.2.1	Requisitos funcionais	49
4.2.2	Requisitos não funcionais	50
4.3	Protótipo	50
4.3.1	Disseminação epidêmica	51
4.3.2	Formato do pacote de transporte e mensagem de aplicação	53
4.4	Diagramas UML	56
4.4.1	Diagrama de Estados	56
4.4.2	Diagrama de Sequência	57
4.5	Dificuldades Encontradas	59
5	RESULTADOS	61
6	CONCLUSÕES	67
7	TRABALHOS FUTUROS	69
	REFERÊNCIAS	71

1 Introdução

Atualmente cerca de 1,25 milhões de pessoas morrem e outras 50 milhões ficam feridas com sequelas em acidentes de trânsito por ano em todo o mundo (SAÚDE, 2015). Apesar desta estatística ter estabilizado desde 2007, os valores ainda preocupam. Uma pesquisa realizada pelo Instituto de Pesquisa Econômica Aplicada (IPEA) sobre os acidentes de trânsito, mostrou que os acidentes nas rodovias nacionais geraram um custo para os cofres públicos no Brasil de 30 bilhões de reais entre o período de janeiro de 2011 a julho de 2012, levando em consideração os danos materiais, os gastos com atendimento e também os prejuízos por interrupção do trabalho (TRINDADE, 2012). Essa pesquisa foi encomendada pelo Departamento Nacional de Trânsito (DENATRAN) e também apurou que as rodovias brasileiras contabilizam, em média, trezentos acidentes por dia, sendo um atropelamento a cada duas horas.

Estes números poderiam ser reduzidos se os motoristas envolvidos nesses eventos pudessem se prevenir antecipadamente em determinadas situações. Para tal, os veículos automotores vêm incorporando diferentes avanços tecnológicos que auxiliam a condução do veículo. A maioria dos sistemas são baseados em sensores, que fazem com que o veículo possa detectar sinais no ambiente e transmití-los de diversas maneiras (ALVES et al., 2009).

Os sistemas de comunicação entre veículos formam as chamadas redes veiculares ou *Vehicular Ad-hoc NETWORK* – VANETs. Essas redes podem ser formadas entre veículos (V2V – *Vehicle-to-Vehicle*) equipados com dispositivos de comunicação sem fio denominados *On Board Units* (OBUs) ou entre os veículos e uma infraestrutura fixa localizada nas margens das vias (V2I – *Vehicle-to-Infrastructure*) (SOUSA; SOARES, 2015b). O que diferencia uma rede veicular de outra rede sem fio é a natureza móvel dos nós dessa rede: seus nós são os veículos que trafegam nas vias públicas como, por exemplo, automóveis, caminhões e ônibus em uma rodovia.

A ideia das VANETs é fazer uso, nos veículos, de tecnologias adotadas nas redes sem fio. Dessa forma, os veículos serão capazes de coletar, gerar e analisar dados. Por exemplo, a comunicação V2V é um exemplo de tecnologia móvel projetada com intuito de permitir que automóveis troquem informações entre si diretamente (SILVA, 2010). Segundo as obras de Biswas, Tatchikou e Dion (2006) e Panayappan et al. (2007), as redes veiculares têm a capacidade de resolver os problemas de segurança no trânsito que aparecem de forma habitual nas rodovias.

Uma tecnologia muito difundida para comunicação sem fio é o Bluetooth (IEEE 802.15.1), que permite transmissão de dados a partir de ondas na frequência de micro-

ondas, porém com alcance limitado de poucos metros a algumas centenas de metros. A comunicação dos dispositivos Bluetooth tipicamente é duplex, ou seja, podem tanto enviar quanto receber dados ao mesmo tempo.

Outra tecnologia para comunicação sem fio, mas de médio alcance é o Wi-Fi (IEEE 802.11) que, embora tenha sido inicialmente concebida como uma tecnologia para WLANs em ambientes fechados e com claras restrições da potência de saída (CROW et al., 1997), também tem sido utilizada atualmente em rádio enlaces ponto-a-ponto e ponto-multiponto em redes sem fio nas áreas metropolitanas e rurais, com a utilização de antenas de alto ganho (FELICE, 2005).

O sistema operacional Android é projetado principalmente para dispositivos móveis com telas sensíveis ao toque, para que o usuário possa manipular os objetos virtuais contidos nele. Portanto, no cenário veicular *smartphones* com Android habilitados com tecnologias de comunicação Bluetooth e Wi-Fi, bem como ferramentas como o GPS (*Global Positioning System*) que pode auxiliar como fonte de informação para as VANETs, em conjunto, podem ser uma ótima interface entre aplicações de trânsito e seu usuário.

1.1 Motivação

O desenvolvimento desse projeto foi motivado pela crescente tendência na utilização das redes veiculares com finalidades voltadas para a segurança no trânsito. No geral, as aplicações de segurança têm como objetivo reduzir o número de acidentes de trânsito por meio da troca de informações entre os veículos e compartilhamento com o motorista. Uma abordagem que pode ajudar a evitar tais acidentes é a troca de mensagens entre veículos, em determinadas situações em que se possa indicar o local do evento, o que está ocorrendo ou o que já aconteceu. Por exemplo, em situações onde um veículo se deparou com algum acidente que esteja bloqueando a pista, um sistema automatizado de alerta poderia avisar aos veículos relativamente próximos do local o que ocorreu logo à frente, viabilizando assim maior tempo de resposta para tal situação. Vale ressaltar que, através de uma solução V2V já existente ou da comunicação veicular via radiofrequência, o recebimento dos alertas rodoviários não fica dependente de haver uma cobertura de telefonia/internet na via de condução, ou de alguma outra infraestrutura (V2I).

A utilização de uma aplicação Android, servindo como interface dos alertas recebidos, se justifica pela atual onipresença e portabilidade desses aparelhos móveis. Também, o custo da implantação de um sistema de alertas rodoviários baseado em Android ser menor em relação a interfaces proprietárias e integradas ao veículo (*dashboards*). Dessa forma os motoristas não precisam ficar dependentes de um *dashboard* pouco acessível e com alto custo de implantação. Assim, usando seu próprio dispositivo Android e um OBU de baixo custo contendo uma ponte de comunicação Bluetooth e o *firmware* V2V (via WiFi *mesh*),

os motoristas terão mais comodidade e opções de escolha.

A segunda motivação é o interesse do discente em explorar áreas de conhecimento sobre redes e comunicação sem fio, vinculando esses conhecimentos com uma recente experiência de programação para dispositivos móveis Android, bem como a vontade de trabalhar futuramente com esses campos de atividades (redes e desenvolvimento mobile).

1.1.1 Solução Proposta

Baseado nos cenários apresentados anteriormente, é possível verificar a necessidade de um sistema voltado para a segurança no trânsito, uma alternativa que seja acessível e viável para os motoristas. Nesta linha, a proposta deste trabalho é construir um sistema para trocar alertas rodoviários entre veículos (V2V). O protótipo terá um *firmware* (que a partir daqui trataremos como OBU) para comunicação V2V e um *app mobile* para interface com o motorista.

O *app mobile* será desenvolvido para dispositivos móveis Android (*smartphone* ou *tablet*), consistindo na interface para geração e exibição dos eventos. A comunicação entre o *app* e o OBU (WPAN) se dará através da tecnologia IEEE 802.15.1 (Bluetooth), que consistirá uma ponte *app*-OBU para a transmissão dos dados de eventos rodoviários. Para prover a comunicação V2V, através da tecnologia IEEE 802.11b/g/n (Wi-Fi) será criada uma *wireless mesh*, rede *ad hoc* através da qual os dados serão trocados entre os OBUs (WLAN). Para garantir a rapidez, economia de bateria do dispositivo móvel e escalabilidade do sistema, um protocolo de disseminação epidêmica será projetado para propagar os eventos entre os veículos.

1.1.2 Trabalhos Relacionados

Um dos principais objetivos das redes veiculares é promover a segurança no trânsito. As aplicações relacionadas à segurança como, por exemplo, alertas de possível colisão, exigem que as mensagens sejam transmitidas em até 100 ms, que é considerado o limite de atraso tolerável para a maioria das comunicações de eventos de emergência (XU et al., 2004). Tal tempo limite em um cenário de VANETs deve incluir o envio de uma mensagem crítica, a resposta do dispositivo veicular e a reação do motorista frente a um evento de emergência. Entretanto, há uma escassez no mercado de dispositivos para rede veicular prontos para o usuário final, mesmo que a arquitetura VANET já esteja se tornando parte da nossa realidade.

Os autores Zaldivar et al. (2011) utilizaram o conector OBD-II para coletar informações diretamente do veículo, para uso em aplicações de segurança de trânsito. Entretanto, foi utilizada a rede de telefonia 3G para comunicação dos veículos com um servidor remoto. Visando atender às restrições temporais anteriormente descritas, a utilização do

padrão IEEE 802.11 pode ser feita para prover a rede veicular em si, evitando o uso ou dependência de tecnologias baseadas em infraestrutura de telefonia 3G ou 4G para notificar esse tipo de evento emergencial por meio da internet (BARCELOS et al., 2014).

Seguindo essa linha, os autores González et al. (2008) realizaram testes de comunicação entre veículos utilizando dispositivos com interfaces de rede IEEE 802.11b (Wi-Fi) de propósito geral, mas com alta potência de transmissão e longo alcance com linha de visada. Os testes foram feitos com dois veículos em movimento e mostraram que foi possível realizar a comunicação direta entre eles (GONZÁLEZ et al., 2008), ou seja, sem a necessidade de alguma infraestrutura preexistente na via.

Já no trabalho de Martelli, Renda e Santi (2011), foram realizados testes com o novo padrão IEEE 802.11p, utilizando um *hardware* dedicado NEC Linkbird-MX para a comunicação entre dois veículos. Em outro trabalho que também utilizou o *hardware* NEC Linkbird-MX, os resultados obtidos sugerem que as especificações propostas pelo padrão IEEE 802.11p puderam ser satisfeitas na prática, proporcionando comunicações V2V, V2I e V2X de maneira eficiente (SUKUVAARA, 2012).

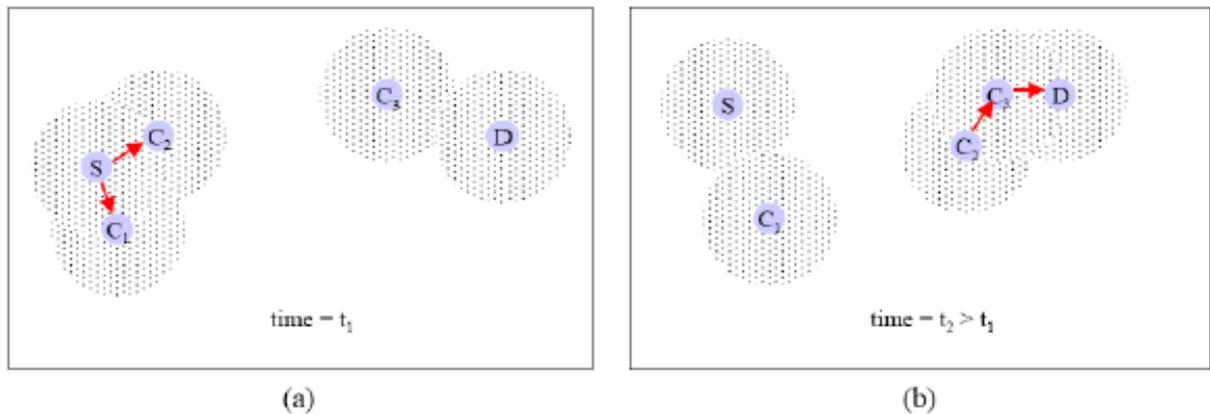
Agafonovs, Strazdins e Greitans (2012) propuseram uma solução que utiliza um *hardware* de comunicação que suporta nativamente o padrão IEEE 802.11p, a placa de comunicação UNEX DCMA-86P2. Em outra avaliação prática, foram utilizados por Teixeira et al. (2013) dois *notebooks* equipados também com a placa UNEX DCMA-86P2, cujos resultados mostraram que o padrão IEEE 802.11p proporcionou uma comunicação eficaz e eficiente entre dois veículos. No trabalho de Barcelos et al. (2014), foi desenvolvido um dispositivo de comunicação capaz de transmitir mensagens no padrão IEEE 802.11p e os resultados mostraram que realizou transmissões em um diâmetro de até 700 metros. Os autores coletaram dados referentes à perda de pacotes e ao atraso na comunicação, para diferentes distâncias.

Sobre o tema da disseminação de mensagens em VANETs, os autores Vahdat, Becker et al. (2000) propuseram um protocolo de roteamento epidêmico (*Epidemic Routing*) no qual as mensagens se difundem na VANET de forma semelhante às doenças biológicas ou vírus. A “infecção” de um OBU se dá pela geração ou recepção de uma mensagem. O OBU infectado então, armazena a mensagem em um *buffer* local para ser capaz de passá-la adiante. Antes do OBU ser infectado com uma atualização, ele permanece em um estado chamado de suscetível, que é quando ele ainda não recebeu a mensagem, mas está sujeito a recebê-la. A “recuperação” de um OBU infectado ocorre quando ele envia a mensagem para uma *Road Side Unit* (RSU), dessa maneira ele se torna “imune” à mesma doença, ou seja, não retransmitirá a mesma atualização novamente.

O algoritmo acima aproveita uma variante da teoria de algoritmos de epidemia. Tal teoria afirma que, dada troca aleatória de mensagens entre os integrantes da rede, even-

tualmente todas as mensagens serão vistas por todos os integrantes, em uma quantidade limitada de tempo. Na Figura 1 é possível ter uma ideia do funcionamento do protocolo de roteamento epidêmico, onde um nó fonte S deseja transmitir uma mensagem para o destino D sem existir uma conexão direta entre eles.

Figura 1 – Comunicação do protocolo de roteamento epidêmico



Fonte: Vahdat; Becker, 2000

Em um trabalho mais recente, Sousa e Soares (2015a), trazem uma proposta onde os veículos transmitem estimativas de congestionamento para seus vizinhos através de *beacons* (dispositivo de comunicação que emite sinais intermitentes por meio de tecnologia *Bluetooth Low Energy*). Tais informações são dados de interesses locais, e os *beacons* formam *clusters* para disseminar entre si os dados inferidos sobre congestionamento.

Neste trabalho foi desenvolvido um protótipo de dispositivo para comunicação veicular de baixo custo e utilizando componentes acessíveis, *firmware* capaz de disseminar mensagens entre OBUs sobre eventos rodoviários de forma epidêmica, através de uma rede WiFi *Mesh* (o conceito de *mesh* será mostrado posteriormente neste trabalho). Dessa forma este projeto trás uma alternativa para a comunicação em VANETs utilizando o protocolo IEEE 802.11b. Além do *firmware*, foi desenvolvido um protótipo de aplicação para *smartphones* com Android, responsável por interfacear os dados para o motorista (recebimento e envio de novos eventos).

1.2 Objetivos

1.2.1 Objetivo Geral

Projetar um protótipo que seja capaz de rapidamente disseminar (enviar e receber) alertas rodoviários entre veículos (V2V), através do OBU construído e tecnologias *wireless*; incluindo uma interface para apresentar esses alertas ao motorista através de uma aplicação *mobile* Android, de maneira objetiva e intuitiva.

1.2.2 Objetivos Específicos

- Familiarizar-se com protocolos para VANETs e definir o OBU para realizar a comunicação V2V sem fio;
- Projetar e implementar o protocolo para disseminação epidêmica de informações;
- Definir a plataforma de desenvolvimento móvel e implementar o *app* para sistema Android;
- Prover uma interface para envio e recebimento de eventos rodoviários, bem como operar o OBU através do *app*;
- Mensurar o desempenho do protótipo na comunicação *app*-OBU e disseminação OBU-OBU.

1.3 Organização do Texto

Este trabalho está dividido em seis capítulos. O capítulo 1 aborda o estado atual do tema, a motivação para o desenvolvimento da ideia, mostra alguns trabalhos relacionados, a solução proposta e também os objetivos do projeto.

O capítulo 2 contém a fundamentação teórica, onde são mostrados conceitos básicos sobre as tecnologias e assuntos envolvidos neste projeto, apresenta conceitos sobre redes móveis sem fio, arquitetura das redes veiculares e aplicações, disseminação de dados nas VANETs, além das tecnologias utilizadas no desenvolvimento do protótipo, como, ferramentas *mobile* e de comunicação sem fio.

O capítulo 3 descreve a metodologia utilizada no desenvolvimento, os materiais que compõem o protótipo do OBU, responsável pela comunicação V2V. Este capítulo também detalha os métodos abordados para a elaboração do projeto.

O capítulo 4 contém informações sobre todo o projeto, nele está descrito o funcionamento completo do protótipo, mostra diagramas UML de estados e sequência do sistema e também relata as dificuldades encontradas no desenvolvimento.

O capítulo 5 descreve os testes realizados para medir o desempenho do protótipo em relação ao ambiente proposto, relata os resultados encontrados por meio da realização desses testes e também expõe as conclusões após o desenvolvimento deste trabalho.

Por fim, no capítulo 6 são descritas as ideias para trabalhos futuros e também exhibe algumas melhorias que poderiam ser feitas no protótipo.

2 Fundamentação Teórica

2.1 Redes móveis sem fio

Uma rede móvel sem fio é qualquer topologia de rede cujos dispositivos (nós da rede) possuem algum tipo de mobilidade, podendo existir alguns nós fixos. A grande vantagem de uma rede sem fio é que ela pode ser implantada com um baixo custo se compara às redes cabeadas. Uma das formas de classificação de uma rede móvel sem fio é quanto às características da mobilidade de seus nós, podendo ser infraestruturada ou em modo *ad hoc* (SILVA, 2010). Na classificação infraestruturada, toda a comunicação entre os dispositivos da rede acontece através de nós fixos, ou seja, das infraestruturas que são responsáveis pela operabilidade da rede.

Já uma rede móvel *ad hoc* (*Mobile Ad Hoc Network* – MANET) é composta por dispositivos capazes de se comunicarem diretamente entre si. Essa comunicação é possível até mesmo quando esses nós estão distantes. Quando isso ocorre, outros nós intermediários são utilizados para rotear os pacotes até o nó de destino, numa estrutura em malha — mais conhecida como *mesh* (AMARANTE et al., 2015). Em MANETs, os protocolos de roteamento encaminham os pacotes pela rede com a finalidade de estabelecer uma melhor rota entre as entidades comunicantes.

Em seguida, serão discutidos alguns conceitos de arquiteturas, padrões e aplicações das redes VANETs, um tipo especial de rede MANET adequada para conectar veículos.

2.1.1 *Vehicular Ad-hoc Network* (VANET)

As redes veiculares (VANETs) são um tipo específico de redes móveis, onde a maior característica é a capacidade dos veículos comunicarem entre si (SILVA, 2010). Essa comunicação *ad hoc* é feita com o auxílio dos veículos que são capazes de receber, armazenar e encaminhar pacotes pela rede. A ascensão dessas redes é devido aos recentes avanços na indústria automobilística que visam agregar novas tecnologias aos veículos, para prover ao motorista acesso a serviços como informações sobre as condições do trânsito e o ambiente (MACEDO et al., 2013).

Para compor uma rede veicular, os nós precisam estar equipados com algum tipo de dispositivo sem fio, que seja capaz de se comunicar com os dispositivos presentes em outros nós, para receber e transmitir pacotes pela VANET. É importante ressaltar que as VANETs trazem características peculiares para o contexto das convencionais redes móveis *ad hoc* (MANETs), dentre essas características destacam-se as seguintes (ZHAO; CAO,

2008):

- **Alta mobilidade** dos nós da rede – os nós estão sempre em movimento e se deslocando em velocidades elevadas;
- **Limitação a estradas e avenidas** – os nós tendem a seguir o trajeto estabelecido pelas vias e sinalizações de trânsito;
- Constantes **mudanças de trajetória** – por estarem limitados às vias, os nós nem sempre conseguem tomar o caminho mais curto até seu destino, sendo obrigados a trocarem de via em algumas situações;
- **Curto tempo de contato** – principalmente, se os nós estiverem se movendo em direções contrárias;

2.1.2 Arquiteturas para VANETs

A arquitetura das redes veiculares representa a forma como os nós se organizam e se comunicam. Atualmente, existem três tipos de arquiteturas principais: *ad hoc* puro (*Vehicle Ad hoc NETWORK* – VANET), infraestruturada ou híbrida.

2.1.2.1 Arquitetura V2V

Na arquitetura *ad hoc* os veículos trabalham como roteadores e transferem dados através de múltiplos saltos, isso porque eles comunicam-se sem suporte externo e sem elemento centralizador (ALVES et al., 2008). Essa arquitetura também é conhecida por V2V (*Vehicle-to-Vehicle*) e, apesar de possuir baixo custo, apresenta desvantagem na conectividade da rede pois depende da densidade e da mobilidade dos veículos.

2.1.2.2 Arquitetura V2I

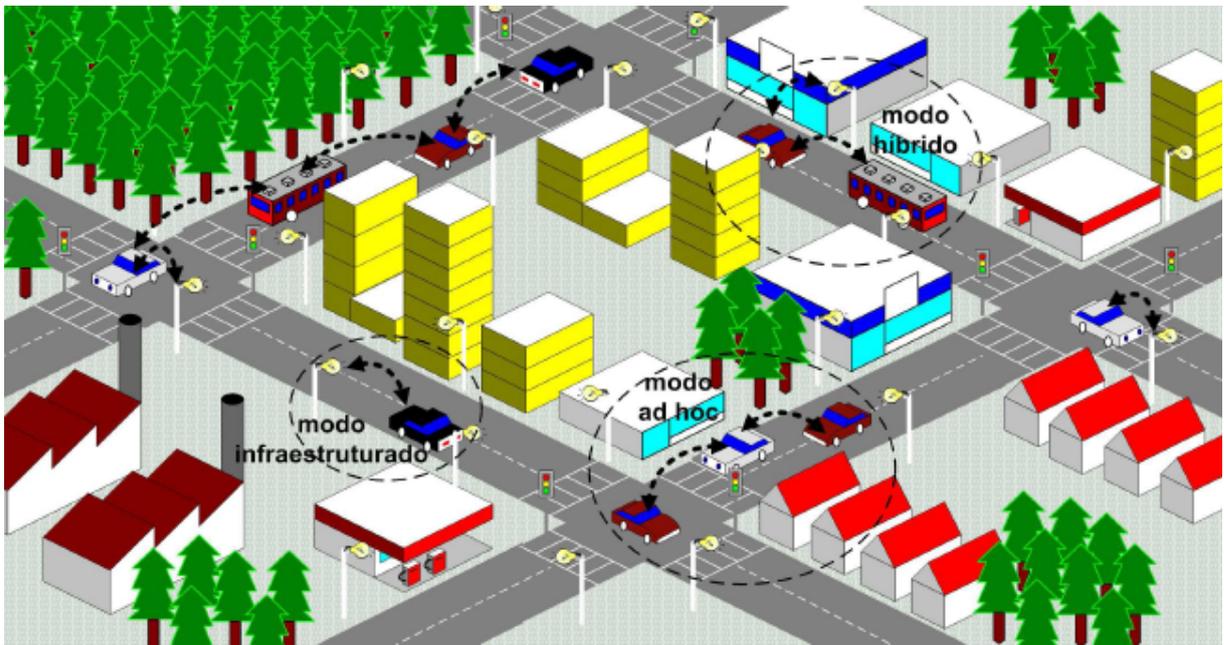
A arquitetura infraestruturada, também conhecida por V2I (*Vehicle-to-Infrastructure*), consegue suprir essa desvantagem pelo fato de serem instalados nós fixos distribuídos ao longo das ruas ou estradas, ou seja, infraestruturas. Tais nós fixos também podem ser chamados de RSUs, funcionando como pontos de acesso (*Access Point*) de rede. Esses pontos de acesso centralizam todo o tráfego da rede e exercem a função de nós intermediários das comunicações (ALVES et al., 2009). Porém, a vantagem de maior conectividade proporcionada traz consigo a desvantagem de um alto custo na implantação desta infraestrutura de rede.

2.1.2.3 Arquitetura V2X

Por fim, a arquitetura híbrida é uma solução intermediária que utiliza tanto a comunicação V2V quanto a V2I. Assim, de acordo com Sousa e Soares (2015b), a chamada

comunicação V2X tenta utilizar o melhor de ambas arquiteturas. A Figura 2 apresenta as diferentes arquiteturas das VANETs. Ela mostra a comunicação entre veículos (V2V), veículos e infraestrutura (V2I), que no caso estão representadas pelos postes (RSUs) e também a comunicação híbrida (V2X).

Figura 2 – O cenário de uma rede veicular



Fonte: Alves et al., 2009

2.1.2.4 Arquitetura WAVE

A reserva de uma faixa de comunicação de curto alcance dedicada (DSRC – *Dedicated Short Range Communications*) pode ser considerada uma das primeiras iniciativas de padronização para a comunicação veicular. Essa iniciativa veio dos Estados Unidos em 1999, por meio da FCC (*Federal Communications Commission*) que alocou 75 MHz do espectro de frequência na faixa de 5,9 GHz para aplicações DSRC. Essa faixa é restrita em termos das aplicações e tecnologias utilizadas, porém não é cobrada taxa pela sua utilização (ALVES et al., 2009). Logo após, surgiram outras iniciativas com o mesmo objetivo em outros países como, por exemplo, na União Européia onde foi alocada a banda de 5,8 GHz para tal fim (JIANG; DELGROSSI, 2008).

Em 2004, um grupo de trabalho da IEEE iniciou a padronização das comunicações em redes veiculares. O padrão IEEE 802.11p é conhecido como WAVE (*Wireless Access in the Vehicular Environment*) e define as camadas físicas e de controle de acesso ao meio (MAC) para redes veiculares, sendo baseado no padrão de redes locais IEEE 802.11a que opera em uma faixa de frequência de 5 GHz, próxima à alocada para as redes veiculares

Tabela 1 – Nomenclatura resumida da arquitetura WAVE

Dispositivo WAVE	Dispositivo que implementa a subcamada MAC e a camada física de acordo com o padrão WAVE
Unidade de Bordo (On Board Unit - OBU)	Dispositivo WAVE móvel capaz de trocar informações com outras OBUs ou RSUs
Unidades de Acostamento (Road side Unit - RSU)	Dispositivo WAVE estacionário que suporta a troca de informações com OBUs
WBSS (WAVE Basic Service Set)	Conjunto de estações WAVE consistindo de um provedor de WBSS e zero ou mais usuários de WBSS
WSM (Wave Short Message)	Mensagem curta WAVE, enviada pelo protocolo WSMP
Provedor de WBSS	Dispositivo iniciador de um WBSS ou emissor de WSMs
Usuário de WBSS	Dispositivo associado a um WBSS ou destinatário de WSMs

Fonte: Alves et al., 2009

(5,850–5,925 GHz). A arquitetura WAVE também designa uma família de padrões (IEEE 1609) que não se restringe às camadas MAC e física.

Os documentos da família IEEE 1609 definem um conjunto de padrões para a comunicação em ambientes veiculares e comunicação sem fio. Segundo Alves et al. (2009) o principal objetivo do IEEE 1609 é prover um conjunto padronizado de interfaces para que diferentes fabricantes de automóveis possam propiciar comunicação entre veículos (V2V) ou entre veículos e infraestruturas (V2I). O padrão também considera que os veículos se movem a altas velocidades e que, por isso, as comunicações devem ser feitas brevemente. A Tabela 1 explica a nomenclatura empregada nos padrões da família IEEE 1609, já a Tabela 2 resume os padrões e seus objetivos.

Segundo Uzcátegui, Sucre e Acosta-Marum (2009), dentre os padrões apresentados na tabela 2 podemos dar significativa importância a quatro pertencentes à família IEEE 1609:

- IEEE P1609.1 – Gerente de Recursos: especifica serviços e interfaces da aplicação de gerenciamento de recursos da arquitetura WAVE. Descreve também os componentes básicos da arquitetura e define as mensagens de comando, os formatos para armazenamento dos dados e especifica os tipos de dispositivos que podem ser suportados por um OBU;
- IEEE P1609.2 – Segurança: define formatos e processamento seguros de mensagens utilizados pelo sistema WAVE e DSRC. Descreve as funções necessárias para suportar mensagens seguras e a privacidade dos veículos;
- IEEE P1609.3 – Serviços de Rede: especifica os serviços das camadas de rede e de transporte, incluindo o endereçamento e o roteamento. Outra função desse padrão é a definição da MIB (*Management Information Base*) para a pilha WAVE que suporte pilhas de protocolo IPv6 e o WSMP (*Wave Short Message Protocol*);

Tabela 2 – Lista de padrões que fazem parte da arquitetura WAVE

Protocolo	Documento do padrão	Propósito do padrão	Número da Camada no Modelo OSI
PHY e MAC da WAVE	IEEE 802.11p	Especifica as funções requeridas para as camadas PHY e MAC para um dispositivo IEEE 802.11 trabalhar com a rápida variação de aplicações veiculares.	1 e 2
Arquitetura	IEEE 1609.0	Apresenta uma visão geral da arquitetura WAVE, seus componentes e sua operação.	Nenhum
Gerenciamento de recursos	IEEE 1609.1	Descreve uma aplicação que permite uma interação com o OBU.	Nenhum
Serviços de segurança	IEEE 1609.2	Formato de mensagens seguras e seu processamento.	Nenhum
Serviços de rede	IEEE 1609.3	Serviços de endereçamento e roteamento em sistemas WAVE.	2, 3 e 4
Operação Multicanal	IEEE 1609.4	Provê melhorias na camada MAC do IEEE 802.11p para que ele suporte operações multicanal.	2
Gerenciamento de comunicação	IEEE 1609.5	Define os serviços de gerenciamento de comunicação para conexão wireless entre OBUs e entre OBUs e RSUs.	Nenhum
Serviços de gerenciamento de dados	IEEE 1609.6	Em desenvolvimento, inclui um gerenciamento de transmissão wireless e recursos de álias.	
Troca de dados de pagamento via IEEE 802.11	IEEE 1609.11	Define um nível básico de interoperabilidade técnica para um equipamento de pagamento eletrônico via WAVE	Nenhum
Atribuições de identificadores	IEEE 1609.12	Especifica a atribuição de identificadores WAVE	Nenhum

Fonte: Uzcategui; Acosta-Marum, 2009

- IEEE P1609.4 – Operação de múltiplos canais: define modificações no padrão IEEE 802.11, para a operação em múltiplos canais para a arquitetura WAVE.

Uma vez contextualizado os esforços de padronização da comunicação veicular, é possível seguir adiante e descrever sobre o estado das pesquisas na área.

2.1.3 Aplicações de VANETs

Quando se trata de redes móveis como as MANETs, o consumo de energia normalmente é um fator crítico pois geralmente os nós são dispositivos sem fio alimentados por baterias (SILVA, 2010). Porém, as redes veiculares possuem a vantagem de terem disponível uma fonte de energia praticamente ilimitada, uma vez que as baterias dos veículos são

recarregadas durante seu uso, proporcionando às aplicações pouca restrição em relação ao consumo de energia.

Segundo Sousa e Soares (2015b), As aplicações de redes veiculares podem ser divididas em três classes: segurança no trânsito, entretenimento e assistência ao motorista. Quando se trata de segurança no trânsito, Alves et al. (2009) dizem que o maior desafio é divulgar de forma rápida as informações para que o condutor tenha tempo para reagir. A divulgação dessas informações geralmente é limitada aos nós localizados na proximidade do veículo que emite o aviso. Sobre entretenimento, destacam-se as aplicações que conseguem compartilhar conteúdos multimídia. E as aplicações de assistência ao motorista são aquelas responsáveis por receber informações que auxiliam a condução como, por exemplo, indicar ao condutor como chegar até uma localização específica como um posto de combustível.

Por causa das características distintas das VANETs é possível identificar as limitações de se integrar tecnologias de comunicação para prover ferramentas de auxílio nas vias públicas.

2.1.3.1 Segurança no Trânsito

Um dos principais incentivos ao desenvolvimento das redes veiculares decorre do potencial de aumento de segurança no trânsito. Geralmente essas aplicações têm o objetivo de reduzir o número e a gravidade dos acidentes nas vias através da troca de informações entre os veículos (ALVES et al., 2009). Nessa classe destacam-se aplicações que divulgam informações sobre as condições gerais de ruas e estradas. Ainda segundo Alves et al. (2009), tais informações podem ser exibidas ao motorista ou usadas para acionar algum sistema público de prevenção.

Para evitar os acidentes de trânsito, que podem ser causados tanto por veículos desgovernados como por veículos lentos ou até mesmo parados, mensagens que informem a posição desses veículos podem ser disparadas periodicamente, avisando aos motoristas próximos sobre uma possível situação de risco. Em consequência do caráter de emergência das informações dessa classe, as aplicações envolvidas carecem de baixa latência na comunicação para serem efetivas (AMARANTE et al., 2015). Para se ter uma noção da granularidade de tal latência, o trabalho de Ib et al. (2011) propõe como valores ideais a latência de 100 ms e um raio de 150 metros do acidente.

2.1.3.2 Aplicações de Entretenimento

De acordo com Alves et al. (2009), as aplicações de entretenimento em geral estão voltadas para os serviços de mensagens instantâneas (bate-papos), troca de músicas e filmes e a distribuição de áudio e vídeo.

Na Internet, uma das aplicações típicas e de grande sucesso é o compartilhamento de conteúdo baseado em sistemas par-a-par. Nas redes veiculares, essa arquitetura de aplicação é chamada de carro-a-carro (*Car-to-Car* – C2C) (PRINZ; BROCCO; WOERNDL, 2008). A ideia é semelhante ao protocolo BitTorrent, ou seja, os veículos devem trocar pedaços dos arquivos desejados entre si usando a infraestrutura internet, mas para que isso ocorra os nós devem construir uma rede sobreposta na camada de aplicação, para se beneficiar no processo de roteamento. Na literatura existem várias propostas de sistemas par-a-par para compartilhamento de conteúdo nas VANETs, dentre os quais podemos citar o SPAWN (NANDAN et al., 2005), o CarTorrent (LEE et al., 2007), o CodeTorrent (LEE et al., 2006), o V3 (GUO; AMMAR; ZEGURA, 2005), entre outros. Porém, aplicações de entretenimento não são o foco deste projeto e por isso este assunto não será tratado mais a fundo.

2.1.3.3 Assistência ao Motorista

A principal finalidade das aplicações de assistência ao motorista é auxiliar a condução do veículo, a partir do fornecimento de informações úteis como, por exemplo, aviso de estacionamento (ALVES et al., 2009). Panayappan et al. (2007) criaram uma solução para controle e divulgação de vagas de estacionamento, que divide uma área onde há vagas em pequenas zonas, de forma que cada zona seja gerenciada por uma RSU. Tal RSU controla a posição e o estado das vagas, verificando se há alguma vaga livre próxima ao veículo e alertando ao motorista.

É possível citar outros exemplos de aplicações voltadas à assistência ao motorista, tais como disseminação de informações de vias, controle de tráfego, auxílio a cruzamentos e localização em mapas.

2.2 Tecnologias de Comunicação sem Fio

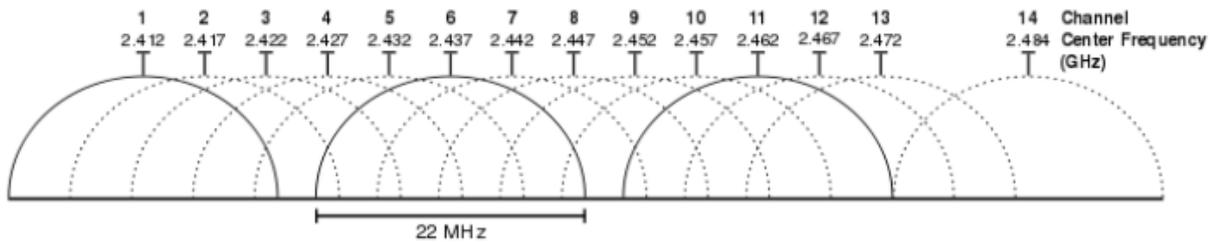
2.2.1 Wi-Fi

Os padrões IEEE 802.11b/g/n, mais conhecidos como Wi-Fi, são padrões de comunicação sem fio para rede de área local (WLAN) que operam na banda ISM (*Industrial Scientific Medical*), utilizando a faixa de frequência de 2,400-2,4835 GHz dividida em 14 canais espaçados em 5 MHz entre si, conforme mostrado na Figura 3.

De acordo com a padronização definida pela IEEE para WLANs operando a 2,4 GHz, os padrões Wi-Fi possuem as seguintes características:

- **IEEE 802.11b:** Largura de banda de 22 MHz, taxa de transmissão de até 11 Mbit/s e com alcance de 35 metros *indoor* e 140 metros *outdoor*.

Figura 3 – Representação da sobreposição de canais na banda de 2.4 GHz



Fonte: Wikipedia, 2017

- **IEEE 802.11g:** Utilizando uma largura de banda de 20 MHz, com uma taxa de transmissão de até 54 Mbit/s e com um alcance de 38 metros *indoor* e 140 metros *outdoor*.
- **IEEE 802.11n:** Oferece uma taxa de transmissão de até 72,2 Mbit/s utilizando uma banda de 40 MHz, com taxa de transmissão de 150 Mbit/s e alcance de 70 metros *indoor* e 250 metros *outdoor*.

Embora originalmente tenha sido concebido como uma tecnologia para comunicação sem fio em ambientes fechados e com restrições da potência de saída, o Wi-Fi pode também ser utilizado para estabelecer rádio enlaces ponto-a-ponto e ponto-multiponto em redes sem fio nas áreas metropolitanas e rurais, com a utilização de antenas de alto ganho (FELICE, 2005).

2.2.2 Bluetooth

O padrão IEEE 802.15.1, mais conhecido como Bluetooth, é um padrão de comunicação sem fio de curto alcance que tem apresentado gradual desenvolvimento nas últimas décadas, em parte devido à sua compatibilidade e eficiência energética. O Bluetooth utiliza ondas curtas de rádio UHF de 2,400-2,485 GHz (SIG, 2017e). Portanto, opera na congestionada banda ISM (*Industrial Scientific Medical*) e, para evitar interferências e garantir compatibilidade, um dispositivo que estabelece uma conexão Bluetooth deve mudar de um canal para outro de forma rápida.

O Bluetooth visa satisfazer aos seguintes requisitos: baixo custo, baixo consumo de energia e tamanho reduzido (PIRMEZ; REZENDE, 2003), permitindo que dispositivos computacionais comuniquem-se entre si em um curto alcance, numa topologia chamada WPAN (Redes de Área Pessoal Sem Fio). Atualmente, o Bluetooth é uma tecnologia bem sucedida e onipresente: encontra-se incorporada em equipamentos portáteis e móveis tais como *notebooks*, *smartphones*, *tablets*, sensores corporais e fones de ouvido, dentre muitos outros possíveis exemplos.

Como a faixa ISM é aberta e pode ser usada por qualquer sistema de comunicação, é necessário garantir que a comunicação Bluetooth não sofra e nem gere interferências. Isso é possível devido ao esquema de comunicação FH-CDMA (*Frequency Hopping – Code-Division Multiple Access*) que faz com que a frequência seja dividida em vários canais (KARNIK; KUMAR, 2000). Os 79 possíveis canais possuem uma largura de banda de 1 MHz entre os quais são realizados saltos com uma taxa de até 1.600 saltos por segundo (RAPPAPORT, 2009).

Tal procedimento é chamado de “salto de frequência” e permite que a largura de banda da frequência utilizada em determinado momento seja muito pequena (1 MHz), diminuindo as chances de interferência entre dispositivos (LEE; SU; SHEN, 2007). Como o Bluetooth usa esse esquema de salto de frequência (AFH), é improvável que haja interferência significativa de outros dispositivos, sejam eles Bluetooth ou não. Dado que os padrões de salto são pseudo-aleatórios, as chances de que outro dispositivo Bluetooth utilize o mesmo padrão de salto e interrompa uma grande quantidade do fluxo de dados é muito baixa. Além disso, outros dispositivos tipicamente transmitem em uma frequência fixa, portanto só poderiam ter um impacto mínimo sobre os dados transferidos usando Bluetooth.

Segundo a SIG (2017e), o alcance dessa tecnologia é dividido na seguintes classes:

- **Classe 1:** consumo de 100 mW (*miliwatts*) com alcance de 100 metros;
- **Classe 2:** consumo de 2,5 mW com alcance de 10 metros;
- **Classe 3:** consumo de 1,0 mW com alcance de 1 metro;
- **Classe 4:** consumo de 0,5 mW com alcance de 0,5 metros;

Dentre esses, os equipamentos de médio alcance (classe 1) suportam alguma das seguintes versões dessa tecnologia:

- **Bluetooth 2:** vazão de 160 Kbit/s com alcance de até 100 metros;
- **Bluetooth 3:** vazão de 700 Kbit/s com alcance de até 100 metros;
- **Bluetooth 4:** vazão de 270 Kbit/s com alcance de 60 metros ou mais;
- **Bluetooth 5:** vazão de 128 Kbit/s com alcance de até 240 metros;

O Bluetooth 2.1 (*Basic Rate / Enhanced Data Rate*) possui como diferencial o suporte a interface de operação do controlador Bluetooth via UART de três fios, bem como taxas de transmissão de dados de 1 a 2 Mbit/s. Já o Bluetooth 3 (*High Speed*) suporta camada de enlace/física alternativa, apresentando taxas de até 25 Mbit/s transportados por um enlace 802.11. O Bluetooth 4 (*Low Energy*), mais conhecido pela sigla BLE conta com maior economia de energia (0,01-0,5 mW), sendo capaz de exigir pouquíssima energia em modo *stand-by*, porém com o dispositivo permanecendo ativo. Por fim, o novo padrão

Bluetooth 5 (*Internet of Things*) possibilita uma vazão de até 50 Mbit/s (duas vezes maior que seus predecessores), um alcance até 4 vezes maior e a capacidade 8 vezes maior de propagar mensagem por *broadcast*, tendo seu desenvolvimento focado em aplicações de Internet das Coisas (IoT). (SIG, 2017b) (SIG, 2017c) (SIG, 2017d) (SIG, 2017a).

2.2.2.1 Exemplos de uso de Bluetooth em VANETs

Existem aplicações inovadoras que utilizam a tecnologia Bluetooth, como uma divisão do Tráfego Rodoviário Canadense, que utiliza os dados coletados dos dispositivos Bluetooth de viajantes através de sensores instalados em ambos os sentidos das rodovias em Calgary para calcular previsões sobre o tempo de viagem e prever congestionamentos para motoristas (CALGARY, 2017). Apesar da maioria dos dispositivos Bluetooth limitarem as transmissões a um curto alcance para prover vazão de alguns Mbit/s com economia de energia, existem versões de dispositivos Bluetooth para comunicação em longo alcance como, por exemplo, o módulo WT41 da BlueGiga, consistindo em um módulo integrado Bluetooth 2.1+EDR de classe 1. Segundo o fabricante, o módulo WT41 consegue prover comunicação Bluetooth com alcance extremo: mais de 1000 metros com linha de visão e uma vazão reduzida de 600 Kbit/s, entretanto podendo chegar a 2,1 Mbit/s em condições ideais (COMPANY, 2017).

2.3 Tecnologias de Posicionamento

Quando falamos de dispositivos móveis, nos referimos na maioria das vezes à *tablets* e *smartphones*. Esses dispositivos executam um sistema operacional e possuem um poder de processamento comparável com o de alguns computadores (AMARANTE et al., 2015). As funções de dispositivos móveis como *smartphones* vão além das funcionalidades típicas de telefones móveis tradicional.

De acordo com Amarante et al. (2015), isso se deve à integração de vários sensores embarcados e o aumento da capacidade de computação e programação, tornando-os uma plataforma sofisticada. Dispositivos móveis modernos incluem GPS, câmera, acelerômetro, sensor de luz ambiente, sensor de proximidade e bússola (FAHMI et al., 2013), dentre inúmeras outras possibilidades. Esses *smartphones* e *tablets* podem se conectar a outros dispositivos utilizando *wireless*, *bluetooth* e redes móveis de telefonia (por exemplo: 2G, 3G e 4G).

Tais recursos podem ser utilizados por aplicativos móveis (*apps*), propiciando a acesso e coleta de informações que auxiliem seu portador como, por exemplo, obtenção de sua localização geográfica, utilizando sistemas de posicionamento geográfico como o GPS.

2.3.1 Global Positioning System (GPS)

Segundo Othman, Wan e Anuar (2011), o GPS é a ferramenta de posicionamento geográfico mais utilizada mundialmente. O GPS funciona baseado na comunicação com satélites geoestacionários. Para essa comunicação, é utilizada uma constelação de 24 satélites enviando informações de localização e marcação de tempo via *broadcast* (LEICK; RAPOPORT; TATARNIKOV, 2015). Quando um receptor GPS recebe informações de no mínimo três satélites, o receptor consegue calcular através da triangulação dos dados e baseado nos tempos de chegada dos sinais, sua posição geográfica atual. Um quarto satélite entra no contexto para fazer a sincronização do tempo, enviando dados referentes a data e hora em que o sinal foi enviado, formando um conjunto de coordenadas geográficas e tempo. Mesmo sendo o sistema de localização geográfica mais utilizado, para fins civis o GPS apresenta um erro médio de 10 a 30 metros na localização (KAPLAN; HEGARTY, 2005).

2.3.2 Assisted GPS (AGPS)

O AGPS é a técnica de localização comumente utilizada em dispositivos móveis. Ela utiliza a localização por GPS auxiliada, quando estiver disponível por uma conexão à internet (AMARANTE et al., 2015). Basicamente, a técnica AGPS constitui-se em:

- Um dispositivo com conexão *wireless* e um GPS;
- Um servidor AGPS com um receptor GPS, com visada para os mesmos satélites que o aparelho (simultaneamente);
- Uma infraestrutura de rede sem fios que consiste em estações base e um centro de comutação móvel;

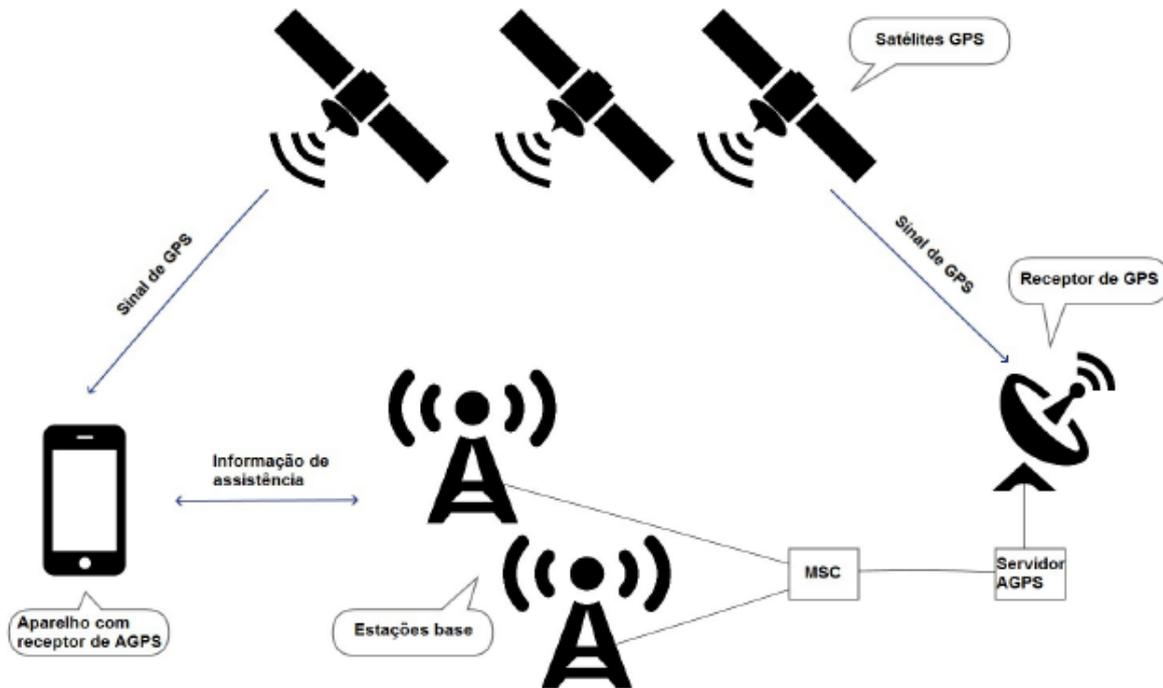
A Figura 4 mostra a estrutura do AGPS:

Essa técnica consegue diminuir consideravelmente o tempo gasto para obter a primeira coordenada, de maneira que o AGPS consegue processar sinais mais fracos que os receptores GPS e possui uma precisão de 15 metros em locais abertos (DJUKNIC; RICHTON, 2001).

2.4 Disseminação de Dados

É muito importante que seja definida a maneira como a informação vai ser disseminada em uma VANET e, para tal, são utilizadas técnicas empregadas em redes DTN (*Delay Tolerant Networks*), que se assimilam as VANETs por apresentarem características parecidas, como por exemplo, conectividade instável (AMARANTE et al., 2015); (PURI; SINGH, 2013).

Figura 4 – Estrutura utilizada no AGPS



Fonte: Amarante, 2015

Na literatura encontram-se diferentes protocolos de roteamento para DTNs mas, segundo Vahdat, Becker et al. (2000), o protocolo epidêmico é o que consegue entregar a informação com menor latência mas, ao mesmo tempo, a rede pode ficar carregada pelo excesso de transmissões de pacotes. O protocolo epidêmico distribui eficientemente as mensagens de uma aplicação entre os nós de sua rede, e esses nós podem retransmitir essas mensagens para outros nós adiante. Dessa maneira, a mensagem tem uma alta probabilidade de chegar ao seu destino através de tais retransmissões, eventualmente (VAHDAT; BECKER et al., 2000).

O mecanismo de difusão de mensagens (*broadcast*) também é muito utilizado no contexto das redes veiculares, em situações onde os nós da rede sem fio se encontrem em um mesmo alcance. Geralmente, as aplicações de VANETs disseminam mensagens pela rede sem que seja necessário conhecer a identificação dos nós que de fato receberão estes dados (OLIVEIRA et al., 2013).

Segundo Chang e Maxemchuk (1984), os protocolos dirigidos à disseminação de informação são divididos em duas classes: protocolos de difusão não confiável e de difusão confiável. Protocolos de difusão não confiáveis propiciam um serviço de melhor esforço, assim eles dão prioridade à disseminação dos dados, sem garantir a entrega dos mesmos a todos os nós existentes na rede, com objetivo de entregar a mensagem ao maior número de nós em um curto período de tempo. Já os protocolos de difusão confiável devem garantir

a entrega da informação a todos os nós pertencentes à rede, utilizando para tal um tempo maior para de fato enviar uma mensagem ponto-a-ponto, tempo esse que é um requisito crítico para algumas aplicações em redes veiculares.

2.5 Eventos Rodoviários

Segundo o (FERREIRA, 1986) evento significa acontecimento. Um evento rodoviário pode então ser classificado como um acontecimento rodoviário, ou seja, pode ser definido como qualquer incidente em uma rodovia ou rua. Na literatura sobre aplicações em VANETs, os trabalhos determinam quais eventos (ou “alertas”) serão abordados. No trabalho de Amarante et al. (2015) são considerados os seguintes eventos:

- Violação de semáforo;
- Luzes de freio de emergência;
- Acidente à frente (Colisão);
- Aviso de mudança de faixa;

Uma outra abordagem para a tipificação de eventos rodoviários consiste em analisar a imensa gama de aplicativos *mobile* (*apps*) existentes para assistência ao motorista. Por exemplo, consideremos os *apps* Waze, Mapas Offline e TomTom Brasil, dentre muitos outros disponíveis na loja de aplicativos (PLAY, 2017). Alguns dos eventos suportados por esses aplicativos são:

- Congestionamento;
- Estrada bloqueada (obstáculos);
- Estrada interditada (Obras);
- Local de interesse;
- Preço de Combustível;
- Presença da Polícia (Blitz);
- Radar de velocidade;

2.6 Desenvolvimento Móvel

Existem inúmeras aplicações para os dispositivos móveis, dentre elas encontram-se aquelas utilizadas como meio de comunicação digital. É notável que o uso de dispositivos móveis como ferramenta complementar está cada dia mais comum, muitas vezes chegando a substituir, alguns nichos profissionais, dispositivos como *desktops* e *laptops*. Além da integração dos *smartphones* com tecnologias de comunicação sem fio (3G/4G, Wi-Fi, Bluetooth), o custo relativamente baixo é uma característica muito forte desses

dispositivos. O processo de desenvolvimento para plataformas *mobile* tipicamente é interativo pois adapta os modelos de *software* com as características de interface desejada (CHARLAND; LEROUX, 2011), que em sua maioria são pré-visualizadas em tempo real.

2.6.1 O Ecossistema Android

Android é o sistema operacional *open source*, ou seja, possui código fonte aberto para que desenvolvedores possam contribuir para a melhoria do Android, bem como identificar e corrigir falhas. O sistema operacional do Android é baseado no *kernel* do Linux (sistema operacional de código livre), que efetua gerenciamento de memória, processos e armazenamento. Por esse motivo, é possível executar vários processos e aplicativos simultaneamente no sistema.

Segundo Lecheta (2016), atualmente o Android está disponível para as mais diversas plataformas, tais como *smartphones* e *tablets*, *smart* TVs, relógios (Android *Wear*), óculos (Google *Glass*), carros (Android *Auto*). Em 2013 foi efetuada uma pesquisa com programadores de dispositivos móveis, na qual foi apurado que 71% dentre todos entrevistados desenvolviam para plataformas Android (MOBILE, 2014). Num levantamento mais atual, foi constatado que o Android é o sistema operacional móvel mais utilizado mundialmente, encontrado atualmente em mais de dois bilhões de dispositivos (C-NET, 2017).

Para desenvolver uma aplicação para Android é necessário ter conhecimentos de determinada linguagem de programação (a linguagem depende da IDE – *Integrated Development Environment*). Porém, após a compilação do código o mesmo é convertido e compactado em um único arquivo “*Android Package File*” que será compatível para qualquer dispositivo que possua o sistema operacional.

O Android SDK é o *software* utilizado para desenvolver aplicações no Android, que possui um emulador para simular o dispositivo, ferramentas utilitárias e uma API completa para a linguagem Java, com todas as classes necessárias para desenvolver as aplicações (LECHETA, 2016).

3 Metodologia

A metodologia foi organizada em oito etapas a fim de atingir os objetivos propostos neste trabalho.

3.1 Revisão Bibliográfica

Foi realizado um levantamento da fundamentação teórica, seguido de pesquisas relacionadas às tecnologias de comunicação e ferramentas de desenvolvimento a serem utilizadas na elaboração do protótipo. Tais estudos tiveram por objetivo prover os conhecimentos necessários para desenvolver o tema abordado e o suporte técnico para o desenvolvimento da solução proposta.

Inicialmente o foco foi no levantamento bibliográfico sobre redes veiculares e suas aplicações. Após, passamos ao levantamento das ferramentas que viabilizaram o desenvolvimento do sistema como, por exemplo, a *suíte* para o desenvolvimento *mobile* (Android Studio), o kit de desenvolvimento para o *firmware* da OBU (nodeMCU ESP-8266) e as tecnologias para rede sem fio de curto e médio alcance (Bluetooth e Wi-Fi). Neste trabalho, foram utilizados materiais bibliográficos publicados em artigos acadêmicos, periódicos e livros, bem como materiais providos pelos fabricantes e consórcios de empresas, tipicamente disponibilizados pela internet.

3.2 Métodos

3.2.1 Definição dos eventos rodoviários a serem suportados

Após o levantamento feito durante toda a pesquisa, foi elaborada a lista de quais eventos rodoviários serão transmitidos e interpretados (alertas ao condutor do veículo) pelo protótipo desenvolvido neste trabalho. Como exposto na fundamentação teórica, a interpretação do conceito de evento rodoviário é vasta, de maneira que a literatura científica propõe diferentes eventos para cada tipo de aplicação em VANETs.

Visando definir um escopo que esteja em conformidade com as capacidades do *firmware* e *app mobile* que compõem o protótipo, neste projeto escolhemos trabalhar com oito tipos de eventos rodoviários:

- Acidente envolvendo veículo(s);
- Animal(ais) na estrada;
- Árvore caída na estrada;

- Buraco na estrada;
- Congestionamento;
- Deslizamento de terra (interditando parte da via);
- Pista interditada (obras);
- Veículo lento na rodovia;

Cada um dos eventos acima, suportados pelo protótipo, pode apresentar uma situação de risco para os motoristas em uma rodovia, situação essa que pode ser evitada caso o motorista seja alertado antecipadamente e tenha tempo hábil para reagir.

3.2.2 Encapsulamento do evento em pacote com formato padronizado

Como foi descrito anteriormente, a comunicação V2V possui uma desvantagem na conectividade da rede por depender da densidade e mobilidades dos nós. Sendo assim, decidimos por não depender de uma infraestrutura prévia de comunicação intraveicular e prover uma rede *wireless mesh* através de nosso protótipo de OBU. Mas, para transmitir os eventos rodoviários entre OBUs, é necessário definir um protocolo de comunicação para nossa aplicação.

Inicialmente, foi necessário projetar o formato do pacote que iria encapsular a mensagem contendo o evento rodoviário. É desejável que essa mensagem seja curta o suficiente para que sua transmissão ocorra de forma rápida e os veículos não percam a comunicatividade antes do envio completo dos novos eventos.

Tal encapsulamento é necessário pois um evento sem informações vinculadas a ele é apenas um dado. Entretanto, quando inserimos características complementares sobre o evento (metadados) em uma mensagem que o transporte, o pacote provê as informações necessárias às tomadas de decisão do *firmware* do OBU, bem como possibilita que o evento seja apresentado de forma compreensível ao usuário através do *app mobile*.

O pacote que transportará a mensagem contendo o evento rodoviário foi projetado para conter os seguintes campos:

- Origem e Destino (opcional);
- Tamanho total do pacote e sua Verificação de integridade (*checksum*);
- Identificador do evento e sua Marcação de tempo (*timestamp*);
- Coordenadas geodésicas (latitude e longitude);
- Dados do evento, contidos na mensagem criada pelo *app mobile*;
- Espaço reservado para uso futuro;

O formato da mensagem (cabeçalho e dados), seus campos e significado serão explicados detalhadamente no capítulo 4, onde será abordada a modelagem completa do

sistema.

3.2.3 Definição do kit para desenvolvimento do protótipo da OBU

Conforme descrito na fundamentação teórica, para compor uma rede veicular os nós precisam estar equipados com algum tipo de dispositivo que seja capaz de se comunicar sem fio com dispositivos presentes em outros nós veiculares. Esses dispositivos de comunicação sem fio veicular são chamados pela literatura de OBUs.

3.2.3.1 DevKit Arduino Pro Mini e módulo nRF24L01+

Num primeiro momento, para a prototipagem da OBU optamos por trabalhar com o *kit* de desenvolvimento Arduino Pro Mini pelo fato de ser um dispositivo barato, com bom suporte da comunidade Arduino e de prototipagem rápida, sendo portanto uma plataforma acessível. O Arduino Pro Mini possui um microcontrolador Atmega328p de 16 MHz e 14 pinos de I/O.

Porém, vale ressaltar que o *kit* Arduino Pro Mini sozinho não seria capaz de realizar a comunicação sem fio com outros nós, a não ser que um módulo (*shield*) de interface de rede seja nele acoplado. Para tal tarefa foi escolhido o módulo de comunicação *wireless* nRF24L01+ que trabalha na frequência de 2.4 GHz e possui 128 canais para comunicação, mas os módulos necessitam estar conectados ao mesmo canal para comunicarem entre si.

De posse de algumas unidades do Arduino Pro Mini e de módulos nRF24L01+, passamos a averiguar se a OBU com este *firmware* atenderia a alguns dos requisitos de comunicação V2V, tais como alcance, latência e taxa de dados. Entretanto, após o desenvolvimento do *firmware* para as primeiras versões do protótipo, com a realização de testes foi averiguado que o módulo nRF24L01+ não se comportava da maneira desejada, provendo baixa taxa de transmissão de dados, curto alcance e sofrendo graves problemas de interferência nos canais (de banda curta, com largura de 1 MHz), o que afetava a qualidade do serviço e a viabilidade da troca de mensagens.

Ademais, o microprocessador do Arduino Pro Mini apresentou uma taxa de processamento incompatível para o desenvolvimento de aplicações de segurança veicular com baixa latência de comunicação. Tais constatações fizeram com que consideremos a dupla Arduino Pro mini & nRF24L01+ uma alternativa inviável para a concretização da proposta deste trabalho.

3.2.3.2 DevKit nodeMCU com microcontrolador ESP8266-12e

Após uma nova análise de equipamentos acessíveis e que poderiam satisfazer os requisitos de comunicação sem fio no cenário de VANET, consideramos o *kit* de desenvolvimento nodeMCU 1.0. Uma característica que chama a atenção é que ele pode ser pro-

gramado em linguagem C/C++ através da própria Arduino IDE, com a qual já havíamos desenvolvido parte do *firmware*, sendo disponibilizadas pelo fabricante várias bibliotecas de apoio, além de ser um *hardware* com preço acessível.

O nodeMCU 1.0 (algumas vezes comercializado como “nodeMCU v3”) possui microcontrolador ESP8266-12e de 80 MHz com 11 pinos de I/O. De acordo com o *datasheet* do fabricante, o ESP8266 suporta os padrões de comunicação sem fio IEEE 802.11b/g/n e também provê a pilha de protocolos TCP/IP integrada (ESPRESSIF, 2017). Mais detalhes sobre os dispositivos utilizados na construção do OBU serão descritos posteriormente na seção de materiais.

3.2.4 Definição da *suíte* de desenvolvimento do *app mobile*: Android Studio

Após o levantamento bibliográfico e análise, foi definido qual plataforma de sistema operacional e quais ferramentas seriam utilizadas para o desenvolvimento do *app*. A plataforma escolhida foi o Android. A ferramenta escolhida foi o Android Studio, IDE (*Integrated Development Environment*) oficial de desenvolvimento para Android. Se comparado com a IDE Eclipse, o Android Studio apresenta alguns diferenciais importantes, de acordo com Lecheta (2016) são eles:

- Editor visual mais fluido e com mais opções;
- Diversas utilidades e facilidades ao desenvolver para Android, sendo muito integrado ao Android SDK;
- *Templates* de projetos para *smartphones*, *tablets*, relógios, TV e carros;
- Atualizações e melhorias frequentes;
- Sistema de *build* mais moderno baseado em *Gradle*;

Ainda segundo Lecheta (2016), a grande diferença entre o Eclipse e o Android Studio é o processo de compilação dos projetos. No Android Studio a compilação é feita pelo *Gradle*, que é um moderno sistema de *builds*, enquanto que no Eclipse o projeto é compilado como qualquer projeto Java dentro da IDE. Por se tratar da IDE oficial do Google para desenvolvimento Android, a gama de materiais de estudo do Android Studio é superior se comparada com outras ferramentas de mesmo propósito. Ainda, a IDE escolhida é gratuita, o que reforçou sua escolha.

3.2.5 Protótipo para exibição dos alertas no *app*

A aplicação *mobile* desenvolvida para interagir com o motorista é responsável por gerar e exibir alertas de eventos rodoviários. A criação de um protótipo de aplicação advém da necessidade de uma interface onde o usuário possa gerenciar os eventos rodoviários, enviando eventos e recebendo notificações de outros.

O *app* desenvolvido consiste em uma aplicação voltada para a segurança no trânsito, por esse motivo a ideia principal é ser intuitivo e bem objetivo, para que dessa forma não tome total atenção do motorista, evitando que ele se distraia do trânsito. Por esses motivos a aplicação evita muito texto e trabalha principalmente com imagens e sons de alertas. A interface do *app* foi construída para que seja possível enviar um alerta de novo evento com apenas um toque.

A Figura 5 mostra a interface do aplicativo desenvolvido.

Figura 5 – Interface da aplicação: a) Tela inicial contendo os eventos; b) Recebimento de evento;



Fonte: O autor

3.2.6 Desenvolvimento do *firmware* para comunicação *app*-OBU

De acordo com Agafonovs, Strazdins e Greitans (2012), são poucos os *hardwares* dedicados de comunicação V2V existentes e, por esse motivo, sistemas de redes veiculares geralmente possuem alto custo. É possível observar, através da seção de trabalhos relacionados, que têm sido desenvolvidos protótipos de sistemas para comunicação em VANET, que tipicamente incluem tanto o OBU quanto a interface do motorista, bem como a ponte para comunicação entre esses dois sistemas.

Neste trabalho, a tecnologia escolhida para fazer a ponte de comunicação WPAN entre o dispositivo Android e o dispositivo OBU foi o Bluetooth, uma solução de baixo custo para comunicação sem fio entre dispositivos portáteis. A tecnologia Bluetooth se encaixa perfeitamente como ponte de comunicação pois tanto os *smartphones* atuais como o OBU desenvolvido neste trabalho possuem suporte ao Bluetooth. Dessa forma, torna-se possível a comunicação entre os dois sistemas (*app*-OBU), utilizando o Bluetooth como intermediário.

3.2.7 Desenvolvimento do *firmware* para comunicação entre OBUs

Para a implementação do sistema de comunicação do OBU foi escolhido o protocolo IEEE 802.11b (Wi-Fi), decisão tomada em função do suporte a comunicação sem fio do microcontrolador ESP8266 integrado ao *kit* de desenvolvimento nodeMCU. O ESP8266 provê a pilha de protocolo TCP/IP com suporte aos padrões IEEE 802.11b/g/n, dos quais o padrão 802.11b utiliza controle de acesso ao meio por espectro espalhado (DSSS), portanto mais robusto na presença de interferências do que os padrões 802.11g/n.

Definida a tecnologia do enlace, decidimos construir uma rede *mesh* (em malha) para a comunicação dos OBUs. A rede *mesh* possui características valiosas para aplicações de redes veiculares, como por exemplo, facilidade de expansão. Isto advém da característica simétrica dos nós, funcionando ora como cliente ora como servidor, provendo assim uma rede par-a-par escalável para uma grande quantidade de nós. Portanto, a rede *mesh* é bem dinâmica, de maneira que seus nós se comunicam entre si sem a necessidade de existência de uma infraestrutura preexistente, ou seja, toda vez que um novo nó se aproxima da rede, ela se expande e o acolhe (CARDOSO; MARQUES, 2012).

3.2.8 Desenvolvimento do protocolo para disseminação epidêmica dos eventos

Mesmo com a escalabilidade proporcionada pela utilização de uma rede *mesh*, é necessário determinar como as mensagens serão roteadas entre os nós da rede. Portanto, deve-se projetar e implementar um protocolo para disseminação dos dados, no caso, especificamente para disseminar os alertas de eventos rodoviários.

Como o objetivo era buscar uma baixa latência de comunicação e ainda assim atingir todos os nós da rede, para essa tarefa foi escolhido desenvolver um protocolo de disseminação epidêmica. Conforme abordado na seção de fundamentação teórica, os protocolos epidêmicos têm como principal objetivo a propagação rápida de dados visando atingir a maior quantidade de nós possível.

O protocolo que foi desenvolvido sorteia dois nós alvo dentre os nós da rede para que sejam “infectados” (com a nova informação). O nó cliente se conecta a cada um dos

dois nós alvo sorteados (servidores) e envia o novo evento a eles. Então, tais nós são considerados como infectados e, por seguirem o mesmo protocolo, cada novo nó infectado escolhe aleatoriamente outros dois nós e assim a disseminação do evento prossegue de forma rápida e escalável. Segundo Coulouris et al. (2013) o número de rodadas para propagar uma única atualização para todos os nós leva $O(\log(N))$ rodadas, onde N é o número de nós. Entretanto, deve-se observar que haverá um certo desperdício de recursos na comunicação entre os nós, visto que pode-se sortear alvos que porventura já tenham recebido a atualização do novo evento, sendo este o *trade-off* a ser pago pela rápida disseminação.

3.3 Materiais

Nesta seção será descrito detalhes sobre os componentes utilizados na elaboração da versão atual do protótipo de OBU, bem como componentes utilizados em versões anteriores dos protótipos elaborados neste trabalho. Abaixo são fornecidas informações básicas mas essenciais para conhecimento dessas tecnologias.

3.3.1 Kit de desenvolvimento Arduino

Arduino é uma placa de desenvolvimento com um microcontrolador ATMEGA, através do qual é possível controlar sensores, motores e equipamentos de radiofrequência, dentre outros componentes. Existem vários modelos de Arduino disponíveis no mercado, dentre os quais serão apresentados a seguir dois modelos que foram utilizados no desenvolvimento inicial deste trabalho.

3.3.1.1 Arduino Mega 2560

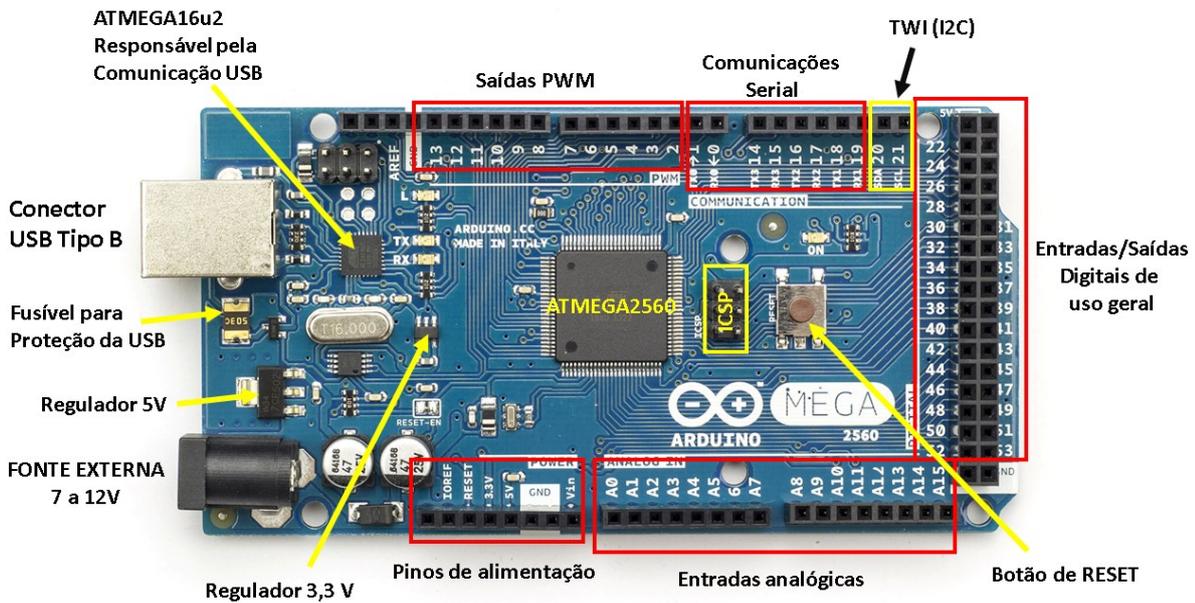
Essa placa é baseada no microcontrolador ATmega2560 de 8 bits e de arquitetura RISC, possui 54 pinos de entradas e saídas digitais, onde 15 desses pinos podem ser utilizados como saídas PWM. Possui 16 entradas analógicas e 4 portas de comunicação serial (SOUZA, 2017). Tal placa também conta com uma quantidade maior de memória em relação a outros modelos de Arduino como, por exemplo, o Arduino Uno.

A Figura 6 mostra os recursos disponíveis e também a pinagem de I/O deste *kit* de desenvolvimento Arduino.

3.3.1.2 Arduino Pro Mini

Essa placa é baseada no microcontrolador ATmega328P, possuindo 14 pinos de entradas e saídas digitais, onde 6 desses pinos podem ser utilizados como saídas PWM.

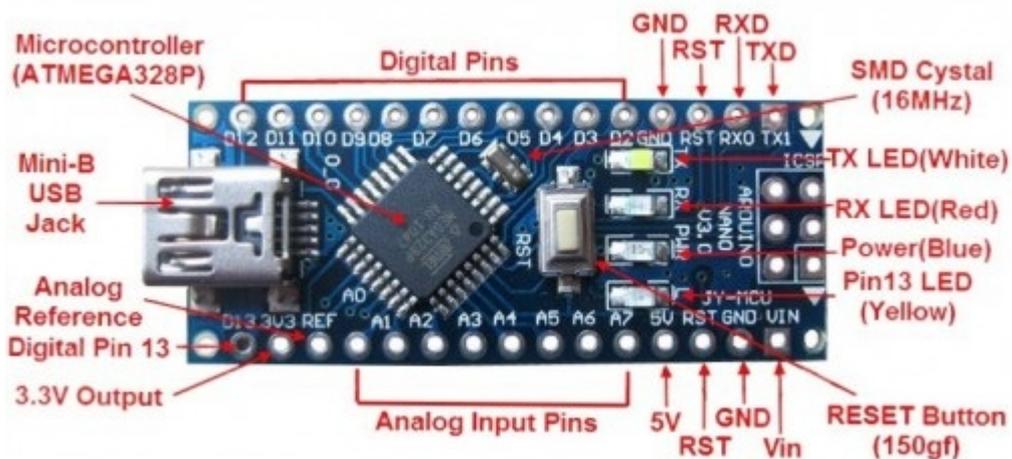
Figura 6 – Pinagem e recursos Arduino Mega 2560



Fonte: Embarcados, 2017

Possui também 8 entradas analógicas. A Figura 7 ilustra a pinagem e recursos deste Arduino.

Figura 7 – Pinagem e recursos Arduino Pro Mini 328P



Fonte: Embarcados, 2017

Tabela 3 – Pinagem nRF24L01+

Pino	Nome	Função
1	VCC	Alimentação
2	VCC	Alimentação
3	CE	Chip Enable RX / TX
4	CSN	SPI Chip Select
5	SCK	SPI Clock
6	MOSI	SPI Slave Data Input
7	MISO	SPI Slave Data Output
8	IRQ	Interrupção
9	GND	Terra
10	GND	Terra

Fonte: Adaptada de Blog FilipeFlop, 2017

3.3.2 Módulo *wireless* 2.4 GHz (nRF24L01+)

O módulo transceptor nRF24L01+ é fabricado pela empresa Nordic e possui uma antena embutida que opera na frequência de 2,4GHz com taxas de transmissão de 250 Kbit/s e de 1-2 Mbit/s. Este módulo possui um conector de 10 pinos e utiliza a interface SPI (Serial Peripheral Interface) o que torna possível interligar o módulo à maioria dos microcontroladores (NORDIC-nRF24, 2017). Outra característica importante do nRF24L01+ é que o mesmo módulo consegue operar como emissor ou receptor, dependendo apenas de uma configuração feita por software.

Figura 8 – Módulo nRF24L01+



Fonte: Blog FilipeFlop, 2017

O alcance de transmissão do nRF24L01+ varia de acordo com o tipo de antena utilizada. Para modelos com antena PCB embutida o alcance varia de 10 metros (indoor) a 50-80 metros em ambientes abertos (outdoor). Porém, existem modelos com antenas LNA externas, esses modelos conseguem atingir um alcance de 100 a 1100 metros em ambientes abertos.

3.3.3 Módulo Bluetooth 2.1 (HC-06)

O módulo HC-06 é uma interface Bluetooth 2.0+EDR Classe 2 (4 dBm), com alcance de 10 a 30 metros e taxa de transmissão de 160 Kbit/s a 2 Mbit/s. O módulo possui 4 pinos: VCC (de 3,6 à 6 V) e GND; RX e TX (até 3,3 V) utilizados para comunicação com o microcontrolador via interface serial TTL. Esse módulo funciona apenas em modo slave (escravo), ou seja, permite que ele se conecte a um outro dispositivos Bluetooth mas não permite que outros dispositivos se conectem à ele, simultaneamente.

Figura 9 – Módulo Bluetooth HC-06



Fonte: Arduino e Cia, 2017

O uso deste módulo na IDE Arduino necessita da biblioteca `SoftwareSerial.h`, que replica a funcionalidade das portas seriais em pinos digitais PWM. É possível ter várias portas seriais “virtuais” com uma velocidade de até 115.200 baud/s cada. Porém a biblioteca `SoftwareSerial.h` possui restrições: caso seja criada mais de uma porta serial “virtual”, não é possível transmitir dados simultaneamente por elas.

Na construção do protótipo deste trabalho de TCC, o módulo Bluetooth HC-06 é utilizado para prover a comunicação sem fio entre o `nodeMCU` e o app mobile de um dispositivo móvel Android, via Bluetooth. As informações recebidas do app mobile pelo módulo HC-06 são repassadas ao `nodeMCU` via serial TTL. Vale ressaltar que antes que a ponte de conexão Bluetooth-TTL possa ser utilizada, é necessário previamente parear o módulo Bluetooth com o dispositivo Android com o qual ele irá se comunicar.

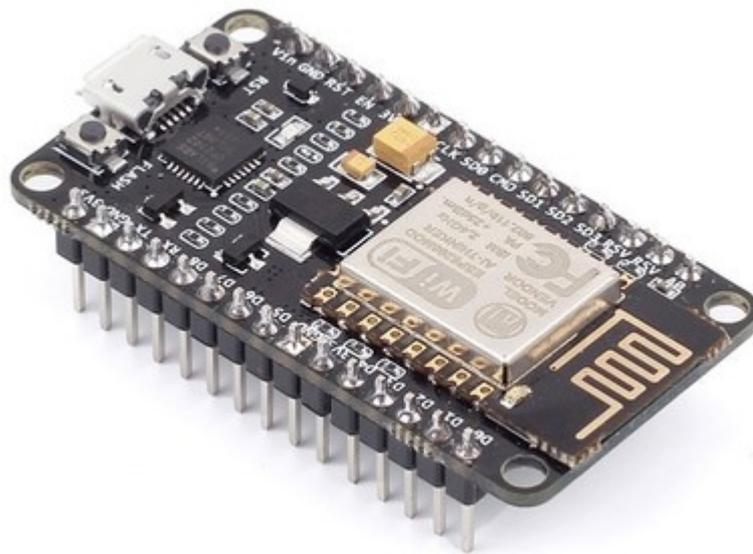
3.3.4 Kit de desenvolvimento NodeMCU 1.0

O módulo ESP8266 é um dispositivo que consiste de um microprocessador ARM de 32 bits com suporte embutido a rede Wi-Fi e memória flash integrada. Para facilitar o desenvolvimento com esse módulo, surgiram projetos de placas que incorporam o ESP8266,

disponibilizando pinos GPIO e alimentação. O kit de desenvolvimento nodeMCU 1.0 foi adotado para o desenvolvimento deste trabalho.

O nodeMCU é uma placa de desenvolvimento que combina o chip ESP8266-12e, uma interface USB-serial e um regulador de tensão de 5 V para 3,3 V. A programação deste *firmware* pode ser feita na linguagem Lua (com a ESPlorer IDE) ou em C/C++ (com a Arduino IDE).

Figura 10 – Kit de desenvolvimento nodeMCU ESP8266-12e



Fonte: FilipeFlop, 2017

O nodeMCU possui um processador Tensilica LX106 de 80 MHz (que pode atingir até 160 MHz), uma memória RAM de 20 KB e uma memória flash de 4 MB. Outras especificações do nodeMCU são: possui conversor analógico digital (ADC); wireless padrão 802.11b/g/n; antena PCB embutida; conector micro-usb; modos de operação STA/AP/STA+AP; suporta até 10 conexões TCP/IP simultâneas e suporta upgrade remoto de *firmware* OTA (Over the Air) (NODEMCU, 2017). A plataforma nodeMCU possui 10 entradas digitais GPIOs operando a 3,3V e uma entrada analógica GPIO a 1,8V.

Além das vantagens de processamento, o nodeMCU em relação ao Arduino também possui vantagens como: baixo custo, suporte integrado a rede Wi-Fi, tamanho reduzido da placa e baixo consumo de energia. Utilizando somente a antena interna embutida, que provê potência máxima de +19 dBm no modo 802.11b, é possível alcançar de 50 a 350 m. Caso uma antena externa LNA seja acoplada, é possível obter alcance de até 500 m ou até mesmo chegar a 4500 m com antena tipo Dish e a baixas taxas de dados (< 1 Kbit/s).

Figura 11 – Pinagem nodeMCU ESP8266-12e



Fonte: Blog FilipeFlop, 2017

4 Projeto

Neste capítulo é documentada a modelagem do sistema desenvolvido, um protótipo para disseminação de eventos rodoviários, com o intuito de alcançar os objetivos definidos anteriormente. Serão apresentados a definição do escopo, a análise de requisitos, diagramas de estados e sequência, as implementações e os testes do protótipo.

4.1 Escopo

Neste trabalho, é proposto um protocolo de comunicação V2V que dissemina eventos rodoviários (ex.: condições adversas da via) de maneira epidêmica, a fim de atingir a maior quantidade de nós possível num tempo razoável, bem como um protótipo de aplicação Android com o intuito de interagir com o motorista, capaz de gerar alertas de eventos e exibir novos eventos recebidos de vizinhos.

Portanto, o escopo deste trabalho trata dos mecanismos de interação entre os dois sistemas desenvolvidos (*app*-OBU e OBU-OBU). O OBU para comunicação V2V foi implementado com um microcontrolador ESP-8266 12e, programado através da IDE do Arduino utilizando a linguagem de programação C/C++, ao passo que a aplicação *mobile* Android foi implementada na IDE Android Studio na linguagem de programação Java.

4.2 Requisitos

Os requisitos mínimos para o funcionamento deste protótipo são:

- Dispositivo móvel que utilize sistema operacional Android com versão a partir da 4.4 (KitKat) com Bluetooth versão 3.0 ou posterior;
- Microcontrolador ESP-8266 12e ou compatível;
- Módulo Bluetooth HC-05 ou HC-06;

A seguir serão apresentados os requisitos funcionais e não funcionais que delimitam o escopo deste projeto e seu funcionamento.

4.2.1 Requisitos funcionais

Aplicação Android (interface com o motorista)

1. O sistema deve ser capaz de ativar o Bluetooth do *smartphone*;

2. O sistema deve ser capaz de exibir os dispositivos Bluetooth que estão pareados com o *smartphone*;
3. O sistema deve permitir a escolha e conexão com um dispositivo Bluetooth previamente pareado;
4. O sistema deve ser capaz de enviar as informações de um evento gerado pelo usuário, via Bluetooth para o OBU do veículo;
5. O sistema deve avisar ao usuário que o evento foi enviado;
6. O sistema deve ser capaz de interpretar e exibir para o usuário, de forma objetiva e intuitiva, um evento recebido do OBU.

***Firmware* OBU** (dispositivo de comunicação sem fio)

7. O sistema deve ser capaz de receber, via Bluetooth, os eventos enviados do *app*;
8. O sistema deve ser capaz de detectar a existência nós vizinhos (outras OBUs);
9. O sistema deve ser capaz de se comunicar com determinado nó vizinho;
10. O sistema deve ser capaz de disseminar de forma epidêmica os eventos recebidos do *app* aos nós vizinhos;
11. O sistema deve ser capaz de receber eventos enviados pelos nós vizinhos;
12. O sistema deve verificar a duplicidade dos eventos recebidos dos vizinhos;
13. O sistema deve ser capaz de continuar a disseminação dos eventos recebidos dos vizinhos;
14. O sistema deve ser capaz de enviar para o *app*, via Bluetooth, os eventos recebidos de nós vizinhos.

4.2.2 Requisitos não funcionais

1. O sistema do aplicativo deve ser desenvolvido em plataforma para dispositivos móveis;
2. O sistema do aplicativo deve ser desenvolvido em Java;
3. O sistema do OBU deve ser desenvolvido em C/C++;
4. O sistema do OBU deve utilizar bibliotecas, disponibilizadas pelo fabricante do microcontrolador, que auxiliam na sua implementação;
5. O protótipo deve ser capaz de fazer a integração da comunicação advinda de ambos os sistemas (*app*-OBU e OBU-OBU).

4.3 Protótipo

Para a realização deste projeto foi desenvolvido um *firmware* (sistema com *software* embarcado em *hardware*) capaz de disseminar eventos rodoviários através de um protocolo

epidêmico, para que alertas desses eventos possam ser exibidos através de um *app* em um dispositivo móvel, auxiliando ao condutor do veículo.

O *hardware* inserido no veículo (OBU), é composto por uma placa de desenvolvimento nodeMCU que possui comunicação via 802.11b integrada (Wi-Fi) e um módulo 802.15.1 (Bluetooth BR/EDR) acoplado. Esse OBU, recebe via Bluetooth dados enviados pela aplicação *mobile*, contendo novos eventos gerados pelo usuário. Também é possível enviar do OBU ao *app* eventos recebidos de nós vizinhos na rede *mesh*. Quando recebe um evento de um nó vizinho, o OBU é responsável por continuar a disseminação do evento rodoviário na rede *mesh*.

4.3.1 Disseminação epidêmica

A disseminação de eventos desenvolvida no protótipo foi baseada nos fundamentos de algoritmos epidêmicos, de maneira que escolhe-se aleatoriamente algum nó que esteja no alcance de comunicação (vizinho) e troca-se atualizações com esse nó, repetindo-se tal procedimento a cada nova rodada. Desta maneira, garante-se que eventualmente todos os nós receberão tal atualização. Porém, cabe ressaltar que haverá um consumo maior de recursos da rede *mesh* pois determinada atualização pode ser recebida mais de uma vez pelos nós. Porém, isto garante uma baixa latência de propagação da atualização, sendo este o grande diferencial da abordagem epidêmica na disseminação de mensagens em redes *mesh*.

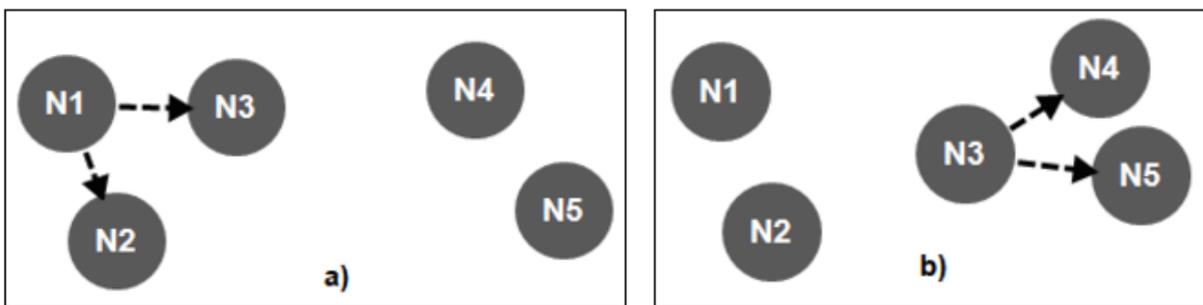
No protocolo de disseminação proposto neste trabalho, foi implementada uma pequena modificação: quando um novo evento é gerado pelo *app*, o nó (OBU) faz uma sondagem de seus vizinhos (*scan*) e sorteia dois nós para infectar, ou seja, enviar o novo evento. Desta maneira, aumenta-se a probabilidade que o novo evento será recebido mais rapidamente pelos nós vizinhos. Quando não há novo evento gerado pelo *app* a ser disseminado, a cada rodada um nó “susceptível” (desatualizado) atua como cliente e procura se conectar a algum vizinho para tentar ser infectado com algum novo evento. Após, passa a se comportar como servidor e tenta receber uma conexão provinda de algum vizinho, com intenção de disseminar a ele o evento anteriormente recebido (do *app* ou de outro vizinho).

Após o recebimento de um evento disseminado através da rede *mesh*, o OBU deve fazer uma verificação de duplicidade desse evento (para diferenciar um novo evento de outro já recebido anteriormente). Essa decisão foi tomada para evitar a irritante exibição na interface do condutor de eventos duplicados. Para identificar evento duplicado, o *firmware* da OBU armazena um *buffer* que contém a origem (ID do transceptor remetente) e o horário de geração do evento (*timestamp* UTC), para os últimos N eventos recebidos. A verificação de duplicidade de um evento recebido é feita comparando-se o ID do nó

originador do evento e horário UTC de criação do evento, com tais dados dos eventos antigos já armazenados no *buffer*. Somente caso o evento recebido seja inédito é que o OBU deve apresentá-lo ao *app* (interface com o condutor do veículo) mas, independentemente de duplicidade os últimos eventos deverão ser propagados adiante senão o processo de disseminação epidêmica será interrompido.

A Figura 12 representa o funcionamento do protocolo epidêmico em determinado momento de sua execução.

Figura 12 – (a) N1 dissemina o evento para N2 e N3; (b) N3 dissemina o evento para N4 e N5



Fonte: O autor

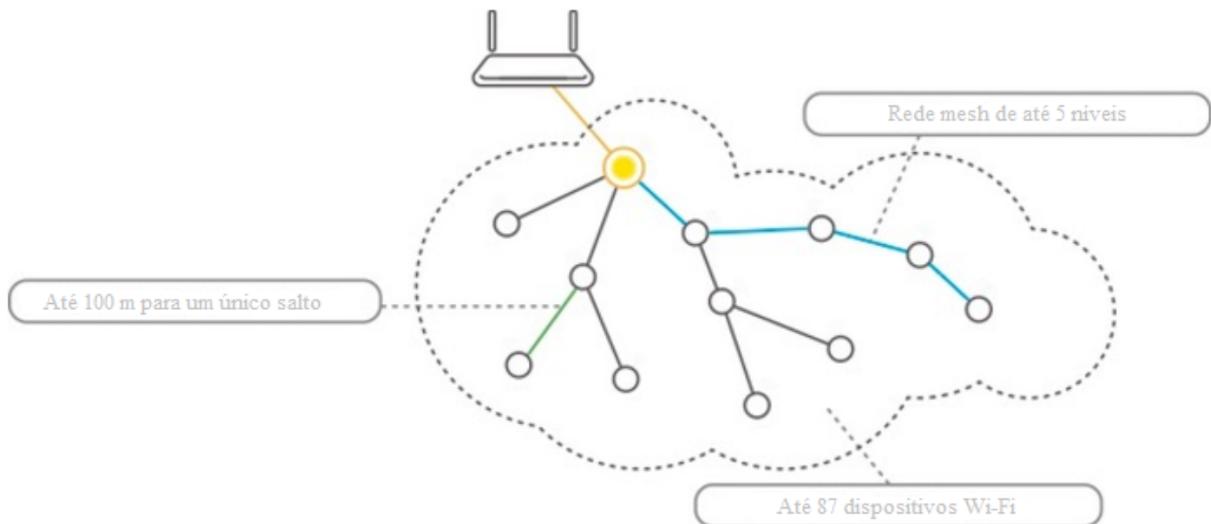
Conforme visto na seção 3.3.4, o microcontrolador ESP8266 12e disponibiliza comunicação Wi-Fi no padrão 802.11b, mais robusto na presença de interferências do que os padrões 802.11g/n. Portanto, a comunicação ponto-a-ponto entre os nós foi realizada através de enlace WiFi 802.11b, com alternância no papel cliente-servidor para o estabelecimento de conexões Wi-Fi, formando assim uma rede *mesh wireless ad hoc*. Conforme descrito na seção 3.2.7, a topologia de rede em malha apresenta características valiosas para aplicações em redes veiculares, como escalabilidade e independência de infraestrutura viária.

Um outro fator que influenciou essa decisão por uma arquitetura de rede em malha foi que o ESP8266 conta com bibliotecas próprias para desenvolvimento de redes *mesh* (WiFiMesh, ESP-Mesh, etc), possibilitando obter um baixo custo para a propagação dos dados e uma prototipagem rápida para o estabelecimento dos enlaces. Entretanto, tal escolha também se fundamenta na capacidade do *hardware* da OBU, pois para utilizar algum protocolo existente para comunicação V2V (ex.: Wave), seria necessário possuímos um *hardware* com suporte ao padrão 802.11p (que funciona na faixa de 5 GHz), implicando no aumento do custo final do protótipo, aumento do tempo necessário pela necessidade de aquisição de novo *hardware* e a própria escassez na oferta de equipamentos desse tipo.

A Figura 13 ilustra as características da ESP-*Mesh*, uma possível arquitetura de rede em malha suportada pelo ESP8266, que se comunica à uma distância de 100 metros

em um único salto, podendo formar uma rede em malha de até 5 níveis e possui capacidade para 87 dispositivos na rede.

Figura 13 – Características da ESP-Mesh



Fonte: ESP-RESSIF Site Oficial, 2017

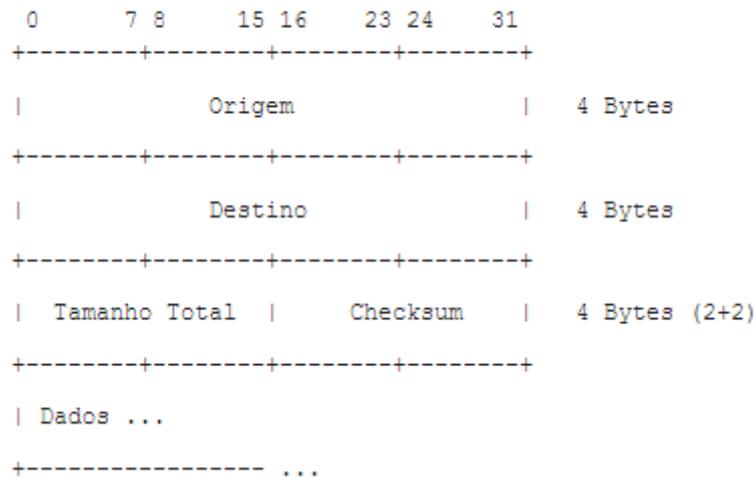
4.3.2 Formato do pacote de transporte e mensagem de aplicação

O pacote de transporte contém as informações necessárias para transportar e interpretar o evento. Como todo pacote de comunicação, consiste em um vetor de bytes estruturado. A Figura 14 demonstra a estrutura do pacote de transporte, de acordo com a posição dos campos do cabeçalho no vetor de bytes.

Como pode ser visto na Figura 14, o pacote de transporte possui quatro campos em seu cabeçalho:

1. **Origem:** A origem é o identificador único do transceptor de origem do evento. Na implementação ela é representada pelo *chip_id*, ou seja, o MAC (*Media Access Control*) do OBU responsável por gerar o evento;
2. **Destino:** O campo destino foi projetado para conter o identificador único do receptor. Porém, atendendo aos requisitos da proposta, o evento é disseminado via *broadcast*, ou seja, ele não possui um nó de destino específico. Sendo assim, na implementação esse campo da mensagem é preenchido com o endereço *broadcast*;
3. **Tamanho Total:** É o tamanho total do datagrama (cabeçalho + dados), projetado para suportar até 8192 bytes;

Figura 14 – Formato do cabeçalho do datagrama do evento



Fonte: O autor

4. **Checksum:** Campo responsável por checar a integridade dos dados. Na implementação o *checksum* é calculado realizando a soma dos dados dos campos do datagrama (cabeçalho e dados);

Conforme pode ser observado pela largura do campo “tamanho total”, o tamanho máximo dos dados transportados pelo pacote (*payload*) foi definido como até 8180 bytes (2^{16} bits - 12 bytes do cabeçalho). Entretanto, quando tal pacote de transporte for transmitido por meio de um enlace 802.11 (Wi-Fi), recomenda-se não enviar *payload* superior a 2300 bytes. Tal valor advém do tamanho máximo de 2312 bytes para o corpo de um quadro 802.11 subtraído de 12 bytes gastos no cabeçalho do nosso pacote de aplicação. Portanto, pode-se enviar dentro deste pacote de transporte um ou mais eventos rodoviários, cuja estrutura da mensagem é apresentada na Figura 15 e explicada abaixo.

Como pode ser visto na Figura 15, a mensagem de aplicação é composta por cinco campos de dados.

1. **Tipo:** O tipo é o identificador único do evento. Na implementação, cada evento recebeu um identificador específico, esse identificador é utilizado para interpretar o significado de cada evento;
2. **Reservado:** Campo extra visando suportar a evolução gradual do protótipo, para suportar situações não previstas no projeto inicial. Portanto, é um campo reservado para usos futuros;
3. **Timestamp UTC:** É o campo que contém a marcação de tempo UTC (*Universal Time Coordinated*), ou seja, o UTC é o “fuso horário” global de referência. A

Figura 15 – Formato do corpo do datagrama do evento



Fonte: O autor

aplicação móvel que é a responsável por preencher este campo do datagrama, uma vez que o dispositivo móvel tipicamente possui receptor GPS e portanto acesso ao *timestamp* UTC;

4. **Latitude:** O campo latitude contém parte da informação de posicionamento global, contendo uma coordenada geodésica do evento. Na geração do evento, o *app* obtém essa coordenada através do receptor GPS do dispositivo móvel;
5. **Longitude:** O campo longitude contém a outra parte da informação de posicionamento global, que descreve a localização de um lugar medido em graus. Esse valor também é adquirido através do GPS do dispositivo móvel;

Para implementar tais formatos de pacote e mensagem, foi declarado no *firmware* um vetor de bytes para receber esses dados. A modelagem desse vetor foi feita da seguinte forma:

- Vetor de bytes com 28 bytes (12 bytes do cabeçalho + 16 bytes de uma mensagem);
- Cada campo do datagrama é representado dentro desse vetor, posicionado de acordo com seu tamanho;
- Os dados são organizados dentro do vetor de acordo com a posição definida pelos formatos do pacote de transporte e mensagem de aplicação.

Quando o *app* vai gerar um evento, ele coleta as informações necessárias e preenche um vetor de bytes que será transmitido, via Bluetooth, para o OBU. Para evitar problemas de representação de dados, os dados a serem enviados pelo *app* foram apresentados em Java (*app* Android) como bytes no mesmo formato esperado em C (*firmware* OBU), através

da biblioteca Javolution. O OBU faz a recepção da mensagem via Bluetooth, através da leitura de byte na interface serial TTL. Então, preenche o cabeçalho do pacote e dissemina o evento adiante. Caso receba um evento do *app* ou de algum vizinho, a OBU interpreta os bytes do pacote e da mensagem de acordo com o tipo de dado apropriado (*short*, *int*, *long*), tomando as decisões apropriadas. Somente quando é considerado pela OBU como sendo um novo evento é que o alerta será exibido ao motorista.

Assim, um evento será disseminado na forma de um vetor de 28 bytes (pacote contendo a mensagem). Segundo Alves et al. (2008) quando os carros movimentam-se em altas velocidades, o emprego de pacotes pequenos pode ser mais eficiente que o emprego de pacotes grandes, pois a probabilidade de erro é menor. Isso é necessário, porque o tempo de conexão entre os nós (em movimento) pode ser mínimo, facilitando assim o envio completo se o pacote for pequeno. Entretanto, vale ressaltar que o formato do pacote projetado suporta transportar uma carga com mais de 140 mensagens (eventos diferentes). Caso não haja preocupação alguma com o tamanho máximo do quadro de enlace, cerca de 512 mensagens poderiam ser enviadas por um único pacote de disseminação epidêmica.

4.4 Diagramas UML

A UML (*Unified Modeling Language*) é uma linguagem ou notação de diagramas criada para visualizar, especificar e documentar modelos de *software*, com o objetivo de apresentar visões de diferentes aspectos do sistema. Para facilitar a compreensão do código-fonte que implementa o *firmware* protótipo de OBU, são apresentados.

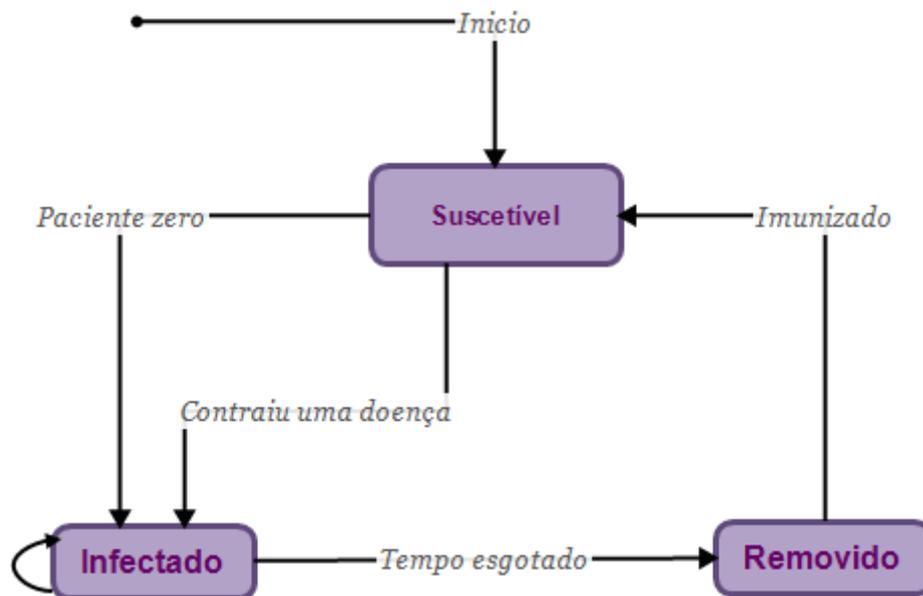
4.4.1 Diagrama de Estados

Consiste na representação do estado de um objeto durante a execução de um sistema, ou seja, descreve o ciclo de vida de um determinado objeto no decorrer do funcionamento de um sistema. A seguir serão apresentados o diagrama de estados da disseminação epidêmica de evento e o diagrama de estados de um nó epidêmico.

Segundo Coulouris et al. (2013), os protocolos epidêmicos possuem três estados: suscetível, infectado e removido. No estado suscetível o sujeito (pessoa ou nó) está passível de receber uma doença (atualização ou evento). Nesse estado, ele pode ser contaminado por um outro sujeito que já esteja infectado ou desenvolveu uma nova doença (paciente zero, que gera uma nova atualização). Quando isso ocorre, o sujeito muda do estado de suscetível para o estado de infectado. Nesse estado ele passa a infectar novos sujeitos, disseminando a doença e só deixa esse estado após determinado tempo. Após atingido um limiar, o sujeito passa ao estado de removido, nesse estado ele descarta a atualização (se imuniza da doença). Essa imunização pode ser tratada de várias maneiras, o sujeito pode já ter infectado um número N de pessoas e, por isso, ter perdido o interesse ou ele pode

simplesmente ter se curado após determinado período de tempo. Quando isso ocorre ele volta ao estado suscetível. Esse processo pode ser observado na Figura 16.

Figura 16 – Diagrama de estados de um protocolo epidêmico



Fonte: O autor

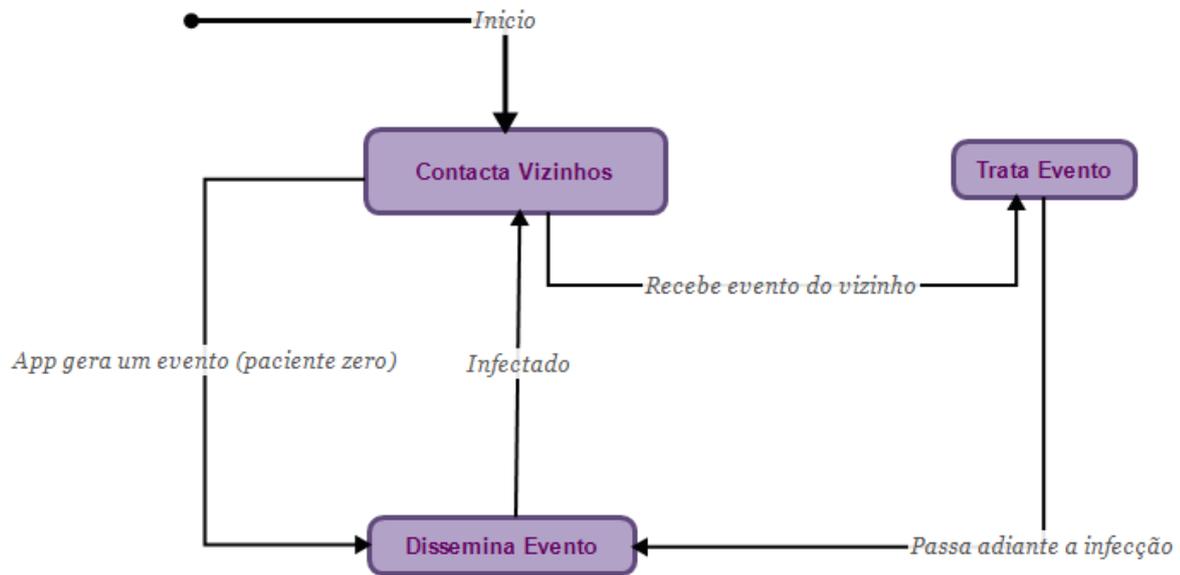
Quando um nó encontra-se no estado “contacta vizinhos” ele está esperando receber uma atualização. Podemos dizer que está suscetível a nova infecção. O nó muda desse estado quando a aplicação *mobile* gera um novo evento (paciente zero) ou quando ele recebe um evento de um nó vizinho (infecta-se). Quando é o *app* que gera o evento, o nó vai para o estado onde ele dissemina o evento, pois foi o paciente zero que gerou determinado evento (nova doença). Porém se o nó tiver recebido evento de um vizinho, além de disseminar a mensagem (infectar outros nós) é preciso tratar tal evento (apresentar alerta ao condutor). Caso tenha recebido um evento repetido, o nó continua a disseminar tal evento até que ele seja irrelevante, porém alternando entre os estados de infectar (dissemina evento) e ser infectado com novas atualizações (conecta a algum vizinho). Conforme mostra a Figura 17.

4.4.2 Diagrama de Sequência

O diagrama de sequência representa a sequência que os processos ocorrem no sistema, ou seja, ele procura representar a sequência em que os eventos ocorrem em um caso de uso. A Figura 18 apresenta o diagrama de sequência do protótipo.

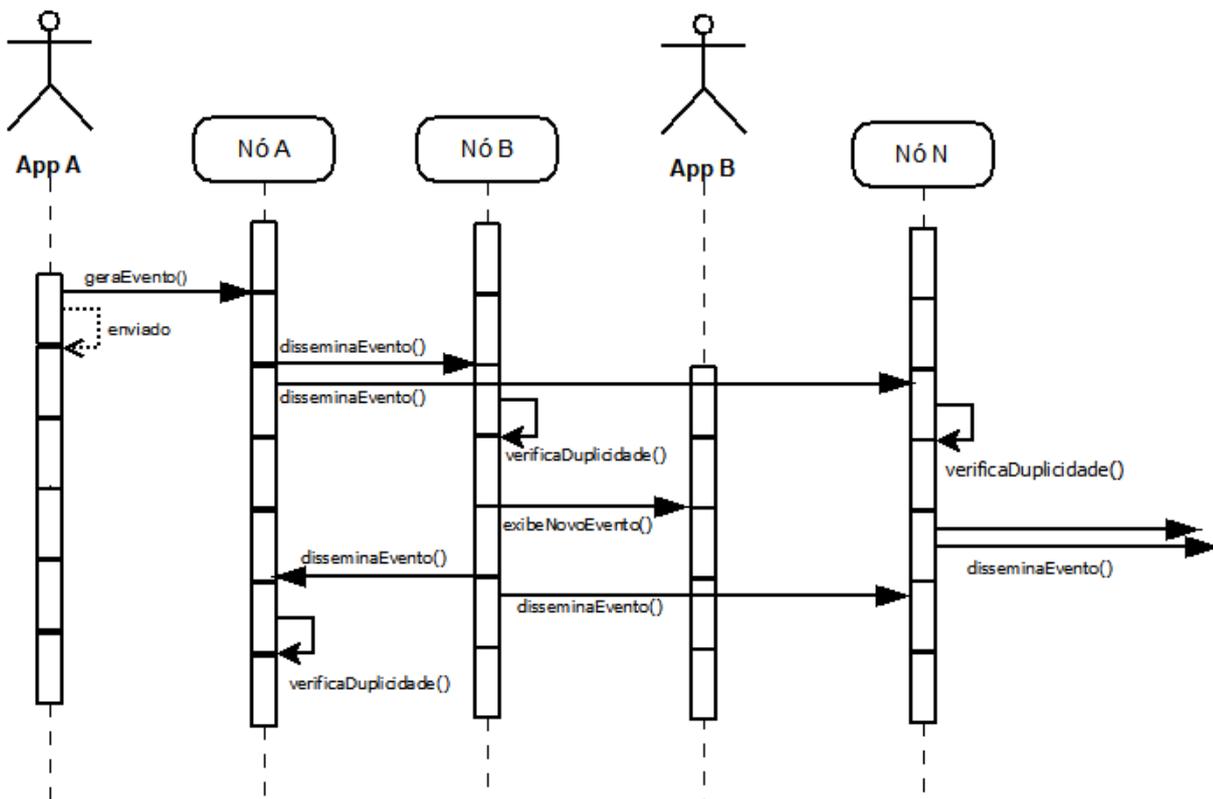
O ciclo de vida dos nós da rede (OBUs) começa no momento em que o veículo é ligado (pois energiza seu OBU) porém, o ciclo de vida do *app* só inicia quando o usuário

Figura 17 – Diagrama de estados de um nó epidêmico



Fonte: O autor

Figura 18 – Diagrama de sequência do protótipo desenvolvido



Fonte: O autor

abrir a aplicação no dispositivo móvel, para que possa visualizar os alertas e comunicar novos eventos.

Num primeiro momento, o *app* gera um evento (conforme escolha do usuário) e transmite uma mensagem para o OBU instalado no veículo do usuário (o *app* e o OBU devem estar pareados via Bluetooth). Quando o OBU recebe esse evento da aplicação ele é responsável por disseminar o evento entre os veículos que estão ao seu alcance (V2V). Para tal, o *firmware* do OBU escolhe aleatoriamente algum vizinho para se conectar e enviar o novo novo evento. Quando o OBU recebe novo pacote de um vizinho, trata-o (apresenta o evento ao *app*) e o dissemina para outros vizinhos, escolhidos aleatoriamente. Este ciclo (rodada) se repete nos N nós da rede.

4.5 Dificuldades Encontradas

No decorrer do desenvolvimento do projeto, foram encontradas algumas dificuldades, especialmente em relação à implementação do *firmware* do protótipo, ou seja, a construção de um OBU funcional para comunicação V2V. Conseqüentemente, muitas mudanças e decisões de projeto tiveram que ser tomadas para alcançar o resultado apresentado neste trabalho.

A primeira dificuldade encontrada foi na utilização do módulo *wireless* nRF24L01+. Apesar de suportar a interface SPI dos microcontroladores, o que é uma vantagem devido a sua grande portabilidade, o autor deste trabalho não possuía prática com tal componente, o que dificultou a montagem do circuito e programação do *software* que operaria a comunicação sem fio. Outro problema encontrado na utilização do nRF24L01+ foi de que este módulo não possui nenhum mecanismo de *feedback* em sua interface, que possa indicar ao usuário que ele está funcionando. Essa limitação de *hardware* atrasou a descoberta de um grave problema no desenvolvimento da primeira versão do protótipo de OBU. A comunicação entre os módulos nRF24L01+ não estava funcionando da maneira prevista, devido a interferências na frequência de comunicação, sofridas de inúmeras fontes que compartilham a banda ISM de 2,4 GHz. Outros dispositivos próximos ao ambiente de desenvolvimento e testes estavam gerando forte interferência e impedindo que os módulos nRF24L01+ trocassem quaisquer informações. Por esse motivo foi tomada a decisão de substituir o *hardware* que proveria o enlace de transmissão para os dados.

Em relação a implementação do código do OBU, houve dificuldades para representação dos dados da mensagem como bytes no pacote. Seguindo a modelagem em camadas popularmente utilizada em redes de computadores, a mensagem contendo informações do evento é representada em um vetor de bytes estruturado, conforme descrito no capítulo 4. Porém, essa representação econômica dificultou a realização de depuração no código. Por se tratar de bytes, existe um problema de apresentação de dados no monitor serial

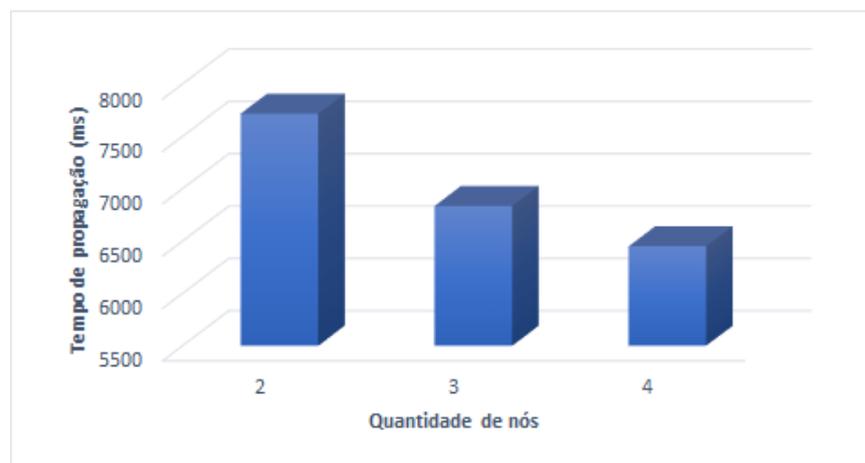
(MS) da Arduino IDE: quando era preciso exibir o conteúdo do evento no MS, o mesmo imprime os bytes desse evento reinterpretados como literais (caracteres) e não como os números referentes ao conteúdo da mensagem, como seria de se esperar. Para resolver o problema descrito acima, foram implementadas funções que fazem a conversão de tipos de dados de/para bytes, sendo necessário manipular os campos do vetor de bytes através do deslocamento de seus bits (*bitshift*).

Ainda, em relação à codificação do protótipo como um todo (*app* e *firmware* OBU), uma outra dificuldade encontrada foi a compatibilização da representação de dados de duas linguagens de programação diferentes em um mesmo sistema. Para que o protótipo opere de forma adequada é preciso que o *app* se comunique com a OBU e vice-versa utilizando uma mesma representação de dados. Como o *app* no Android foi desenvolvido com a linguagem Java e o *firmware* OBU com a linguagem C/C++, foi preciso buscar uma forma única de representação dos dados para que ambos sistemas possam interpretá-los consistentemente.

5 Resultados

Este capítulo tem como objetivo descrever os testes realizados para avaliar o protótipo proposto, e apresentar os resultados encontrados. A disseminação dos eventos através do protótipo foi avaliada por meio da latência de comunicação e densidade dos nós na rede *mesh*. O objetivo foi identificar se o uso deste protótipo para disseminação de eventos em redes veiculares seria viável. Os experimentos foram realizados com 2, 3, 4, 5 e 6 nós na rede *mesh*, limitados apenas pela quantidade de unidades nodeMCU disponíveis. Pôde-se observar que conforme a quantidade de nós na rede aumentava, o tempo de propagação epidêmica do evento diminuía. Vale ressaltar que para os testes de escalabilidade, eventos foram gerados a cada 5 segundos (5000ms). Os valores encontrados nas Figuras 19, 20, 21, mostram através de gráficos e de dados estatísticos, o tempo de propagação junto ao intervalo de envio dos eventos.

Figura 19 – Escalabilidade da rede

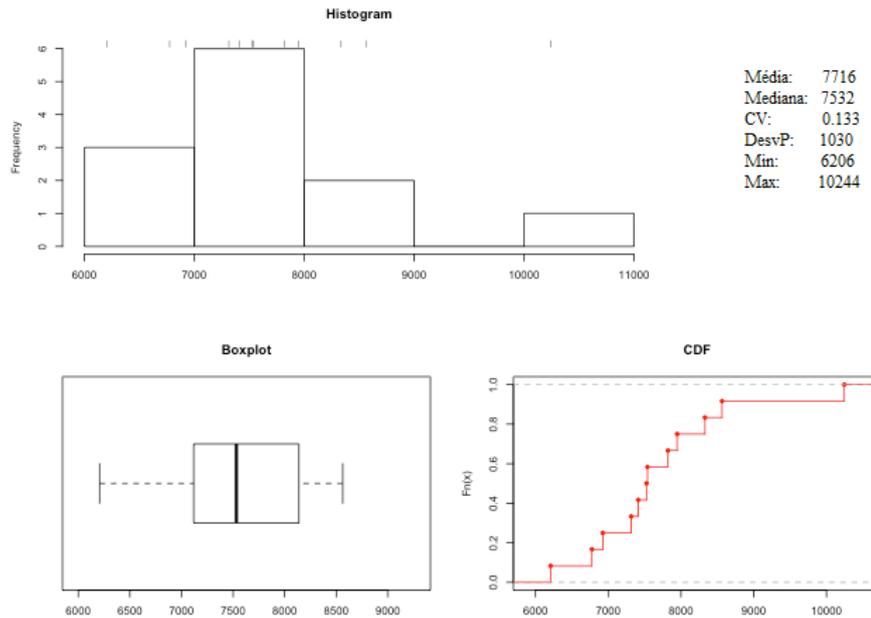


Fonte: O autor

Com o aumento da escalabilidade, a chance de criarem uma conexão cliente/servidor entre dois nós também aumenta, conseqüentemente. No texto abaixo, cada nó da rede é representado pelo *firmware* do protótipo de OBU desenvolvido.

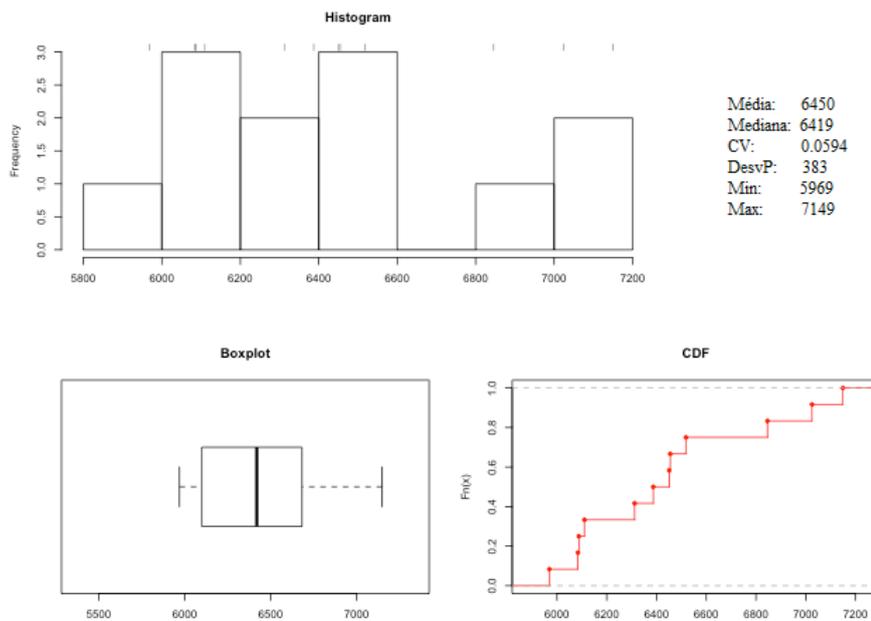
Durante os experimentos, foram mensuradas as latências de diversos aspectos da comunicação entre OBUs. O tempo de envio da mensagem contendo os dados do evento gerado pelo aplicativo *mobile* para o OBU (via Bluetooth 2.1), foi de **23 ms** com um *baud rate* configurado em 9600 bps. Entretanto, caso necessário esta latência poderia ser reduzida aumentando-se a taxa de comunicação serial (TTL-Bluetooth) para o valor máximo suportado pelo módulo Bluetooth HC-06 (1.382.400 bps), o que apresentaria uma latência de aproximadamente 16 μ s para a transmissão de 28 bytes, por exemplo.

Figura 20 – Estatísticas de propagação com 2 nós na rede



Fonte: O autor

Figura 21 – Estatísticas de propagação com 4 nós na rede

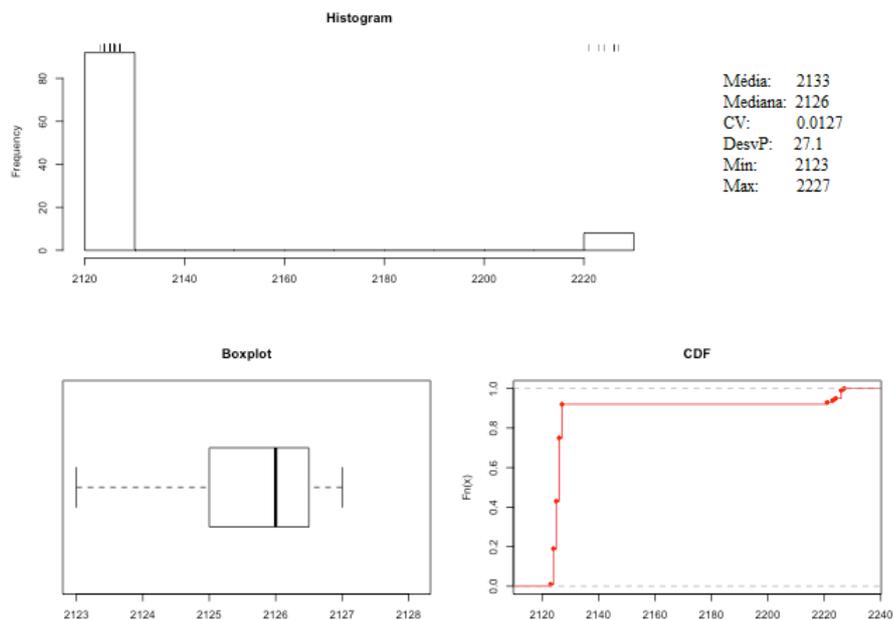


Fonte: O autor

Ao receber um evento do *app*, o OBU realiza uma sondagem (*scanning*) dos vizinhos da rede *mesh* (que estejam em modo servidor, denominado AP) e sorteia um dentre eles para transmitir o novo evento. Tal sondagem é efetuada com uma latência menor que **1 ms**. Após a definição do nó sorteado dentre os APs vizinhos, o OBU deve disseminar o evento a ele. Para isso, necessita primeiramente se conectar ao vizinho e somente então enviar o pacote contendo os dados do novo evento.

Do ponto de vista do nó cliente (modo STA), o tempo gasto para o estabelecimento da conexão foi em média **2133 ms**, com um intervalo de confiança de 95% de que a latência ficaria entre 2126 e 2138 ms. Na Figura 22 é possível observar esses e outros dados estatísticos dos valores encontrados. Pela previsibilidade insinuada pelo estreito intervalo de confiança, atribuímos a causa da magnitude desses valores como devido à utilização, por parte do *driver* WiFi do ESP8266, de um tempo de espera obrigatório para que um nó cliente (modo STA) aguarde o estabelecimento da conexão com o servidor (modo AP). A biblioteca ESP8266WiFi, *driver* WiFi que opera o controlador ESP8266, define uma constante com valor de 1500 (milissegundos) como tempo de espera do cliente para o estabelecimento da conexão com o nó servidor da rede *mesh*.

Figura 22 – Estatísticas de latência de um nó Cliente

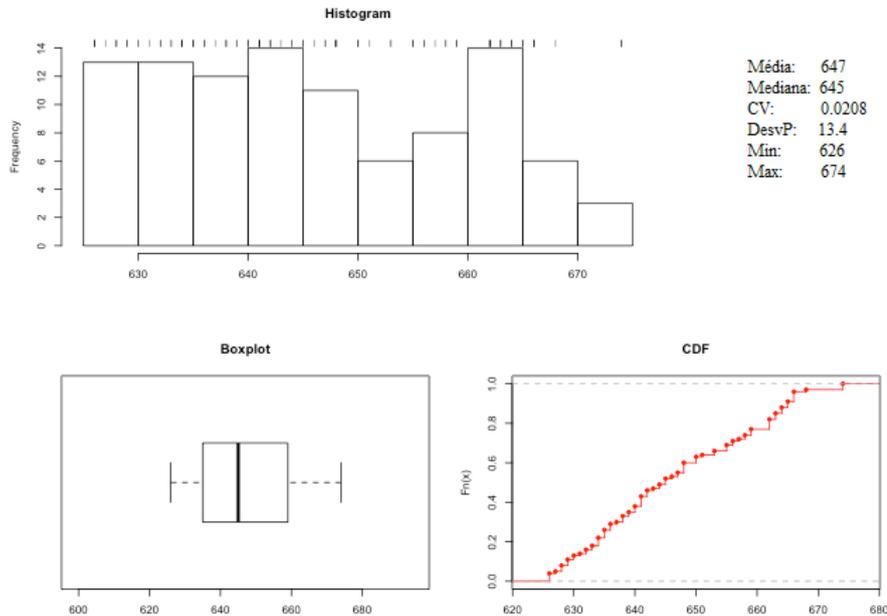


Fonte: O autor

Do ponto de vista do nó servidor (modo AP), o tempo de espera pela conexão de um cliente vizinho apresentou uma latência média de **647 ms** com um intervalo de confiança de 95% de que a latência ficaria entre 643 e 652 ms. Esses dados podem ser observados na Figura 23. Uma vez que o cliente conecta-se ao servidor, os tempos de espera relatados acima poderiam ser interpretados da seguinte maneira: 647 ms + 1486

ms = 2133 ms, ou seja, o servidor recebe a conexão do cliente com 647 ms mas o cliente deve aguardar aproximadamente 1500 ms antes que seu status seja atualizado pelo driver, de desconectado para conectado.

Figura 23 – Estatísticas de latência de um nó Servidor

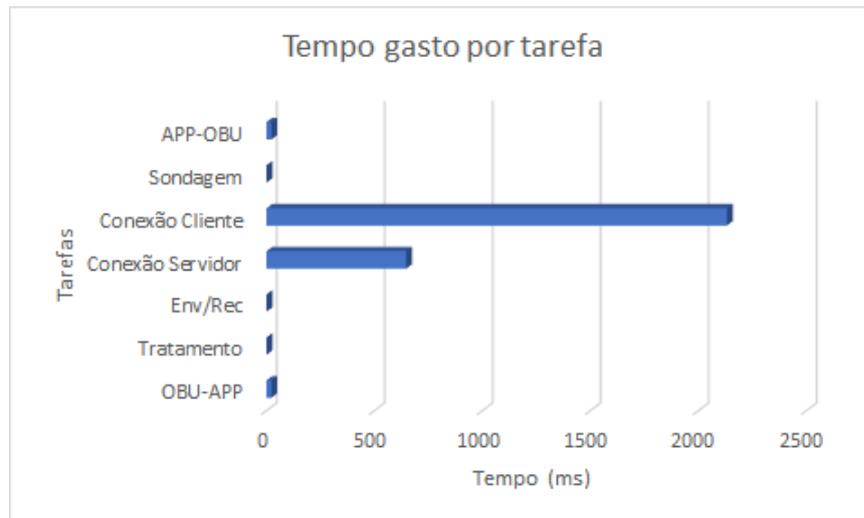


Fonte: O autor

Vale ressaltar que foram conduzidas diversas tentativas de alterar pontualmente o código-fonte do *driver* WiFi do ESP8266 visando reduzir essa espera, entretanto tais modificações invariavelmente inviabilizaram o estabelecimento de qualquer conexão entre os nós. Por este motivo, esta etapa da disseminação de evento ficou com o desempenho prejudicado, conforme pode ser observado através da Figura 24 no gráfico comparativo entre o tempo gasto em cada etapa, o estabelecimento da conexão é a etapa mais demorada de todo o processo. Uma medida paliativa seria tentar reprogramar o comportamento de grandes porções do *driver* WiFi do ESP8266 para agilizar a confirmação do estabelecimento do enlace WiFi ponto-a-ponto, o que não seria viável considerando as restrições de tempo deste trabalho de conclusão de curso.

Uma vez estabelecida a conexão entre os nós cliente-servidor, o tempo de envio/recebimento (entre tais OBUs) de um pacote de 28 bytes contendo o evento é menor do que **1 ms**. Observe que a mensagem contendo o evento tem tamanho de apenas 28 bytes, bem menor que o próprio cabeçalho do quadro 802.11 (34 bytes) e que o enlace 802.11b estabelecido entre os nós suporta taxas de transmissão de 1 a 11 Mbit/s (dependendo da distância entre os nós). Considerando que o quadro 802.11 possuirá 62 bytes no total (34 bytes pro cabeçalho e 28 bytes pros dados), o tempo de transmissão ponto-a-ponto no

Figura 24 – Gráfico representativo do tempo gasto por tarefa no protótipo



Fonte: O autor

enlace 802.11b seria de no mínimo 0,045 ms (a 11 Mbit/s) e no máximo 0,496 ms (a 1 Mbit/s).

Há também uma latência gasta no processamento da mensagem do evento, pois quando o OBU recebe um evento de algum vizinho ele precisa efetuar uma verificação de duplicidade conforme descrito no capítulo 4. Essa verificação consome um tempo inferior a **1 ms**. Caso o nó esteja recebendo um novo evento (e não enviando-o), os dados de origem e UTC (*timestamp*) desse evento devem ser armazenados no buffer que contém os dados de verificação dos eventos antigos recebidos, tarefa que também consome um tempo menor que **1 ms**.

Após a verificação, caso o evento que acabou de ser recebido pelo OBU seja inédito, esse evento deve ser exibido no *app* com intuito de alertar o motorista. O tempo gasto nessa tarefa é o mesmo do recebimento de um evento gerado pelo *app*, pois utiliza a mesma tecnologia Bluetooth 2.1 como ponte de transmissão dos dados.

6 Conclusões

Neste trabalho foi criado um protótipo para disseminar eventos rodoviários entre veículos com o intuito de prover segurança no trânsito. A integração do aplicativo móvel com o OBU possibilitou a comunicação e disseminação dos eventos em uma VANET utilizando os padrões IEEE 802.11b e IEEE 802.15.1.

Através dos experimentos efetuados, foi possível observar empiricamente que o protocolo de disseminação epidêmica obteve eficácia na entrega dos eventos a todos os nós da rede *mesh*. Entretanto, quanto ao desempenho do mesmo, ao considerar as latências de comunicação apresentadas é importante frisar que o protótipo desenvolvido não atenderia aos requisitos de latência de uma aplicação para segurança no trânsito. Conforme mensurado, da perspectiva de quem recebe um evento gasta-se um total de 691 ms e, para quem está enviando o evento, 2177 ms. Porém, para viabilizar o uso do protótipo para fins de aplicações de segurança, o envio da mensagem de alerta não deveria ultrapassar 100 ms, conforme descrito anteriormente na seção 1.1.2. Apesar disso, o tempo de contato entre dois veículos trafegando em sentidos opostos na rodovia em alta velocidade, ainda é maior do que o tempo gasto para enviar uma mensagem. Levando em consideração o alcance da tecnologia Wi-Fi (140 metros), utilizada pelo OBU, valores relacionados ao tempo de contato entre veículos em determinadas velocidades podem ser observado na Figura 25.

Figura 25 – Gráfico representativo do tempo de contato entre dois veículos trafegando em sentidos opostos



Fonte: O autor

Levando em consideração o desempenho atualmente apresentado, o protótipo desenvolvido seria mais apropriado para uso na classe das aplicações para assistência ao

motorista, que são aplicações que não exigem um tempo de resposta imediato do motorista. Ou também para comunicação veicular em ambientes com velocidade controlada, como por exemplo, mineirações e pátios de fábricas.

Este trabalho mostrou resultados que possibilitam uma alternativa para utilização de VANETs através do protocolo IEEE 802.11b, abrindo portas para outras aplicações de comunicação V2V serem desenvolvidas.

7 Trabalhos Futuros

Após o desenvolvimento deste trabalho foi possível enxergar novas ideias de aplicações para segurança no trânsito, como por exemplo, a detecção e alerta automática de acidentes. Mas, para que o mesmo possa ser utilizado para tais fins, deverá ser conduzida melhorias visando a redução do tempo gasto durante a conexão do cliente com o servidor, bem como prover novas funcionalidades e demais aprimoramentos.

No início do desenvolvimento deste trabalho de conclusão de curso, com o intuito de complementar a proposta original, foi realizado uma solicitação via e-mail aos desenvolvedores do popular *app* Waze, para que tivéssemos acesso à API desse aplicativo. A ideia seria fazer uma integração dos eventos gerados pelos milhões de usuários diários do Waze (que utiliza comunicação via internet móvel) com a propagação de eventos rodoviários provida pelo protótipo desenvolvido neste trabalho (que por sua vez não necessita de internet). Entretanto, ainda não tivemos um retorno positivo por parte dos desenvolvedores do Waze, o que inviabilizou tal intento. Portanto, um grande diferencial para versões futuras do protótipo seria concretizar a integração do *app* desenvolvido neste protótipo com a API do Waze, para que os usuários rodoviários tenham acesso a uma gama maior e mais atualizada de eventos, em tempo real.

Sugere-se também integrar um módulo *wireless* de maior alcance ao OBU, como, por exemplo, um módulo nRF24L01+ com antena externa LNA, visando atingir alcances de até 1 Km e interconectar *clusters* distintos pela estrada, cada um com sua rede *mesh* de OBUs.

Melhorias no *software* do OBU também devem ser consideradas como, por exemplo, o aperfeiçoamento do sistema para tratar de questões de segurança como autenticação e criptografia dos dados. No protótipo atual a autenticação e criptografia da rede sem fio foram desabilitadas para reduzir o tempo de estabelecimento da conexão ao enlace 802.11b, já demasiado grande (691 ms). Uma outra sugestão de melhoria seria uma nova versão do protocolo de disseminação, que seja capaz de armazenar não somente o identificador dos eventos já recebidos mas também todo o pacote com o evento, numa cache a ser utilizada para aumentar a probabilidade de que nós recém chegados à rede *mesh* recebam também os eventos mais recentes, funcionando como uma DTN (rede tolerante a interrupções).

Referências

- AGAFONOV, N.; STRAZDINS, G.; GREITANS, M. Accessible, customizable, high-performance IEEE 802.11 p vehicular communication solution. In: IEEE. *Ad Hoc Networking Workshop (Med-Hoc-Net), 2012 The 11th Annual Mediterranean*. [S.l.], 2012. p. 127–132. Citado 2 vezes nas páginas 20 e 41.
- ALVES, R. d. S. et al. Uma análise experimental da capacidade de redes ad hoc veiculares. *SBrTSimpósio Brasileiro de Telecomunicações*, p. 8, 2008. Citado 2 vezes nas páginas 24 e 56.
- ALVES, R. d. S. et al. Redes veiculares: Principios, aplicações e desafios. *Minicursos do Simpósio Brasileiro de Redes de Computadores, SBRC*, 2009. Citado 6 vezes nas páginas 17, 24, 25, 26, 28 e 29.
- AMARANTE, T. C. et al. Detecção automática e alerta de acidentes de trânsito em redes veiculares reais. Programa de Pós-Graduação em Ciência da Computação, 2015. Citado 5 vezes nas páginas 23, 28, 32, 33 e 35.
- BARCELOS, V. P. et al. Sistema de monitoramento de veículos usando dispositivos no padrão IEEE 802.11 p. *XXXII Simpósio Brasileiro de Redes de Computadores e e Sistemas Distribuídos, SBRC*, 2014. Citado na página 20.
- BISWAS, S.; TATCHIKOU, R.; DION, F. Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety. *IEEE Communications magazine*, IEEE, v. 44, n. 1, p. 74–82, 2006. Citado na página 17.
- C-NET. *Google's Android now powers more than 2 billion devices*. 2017. C-net WebSite. Disponível em: <<https://play.google.com/store/apps?hl=pt-br>>. Acesso em: 21 de mai 2017. Citado na página 36.
- CALGARY, C. N. *Wireless waves used to track travel times*. 2017. CTV WebSite. Disponível em: <<http://calgary.ctvnews.ca/wireless-waves-used-to-track-travel-times-1.1054731#>>. Acesso em: 21 de mai 2017. Citado na página 32.
- CARDOSO, T. M.; MARQUES, P. C. F. Rede mesh: topologia e aplicação. *Revista iTEC-Vol. IV*, n. 4, p. 16, 2012. Citado na página 42.
- CHANG, J.-M.; MAXEMCHUK, N. F. Reliable broadcast protocols. *ACM Transactions on Computer Systems (TOCS)*, ACM, v. 2, n. 3, p. 251–273, 1984. Citado na página 34.
- CHARLAND, A.; LEROUX, B. Mobile application development: web vs. native. *Communications of the ACM*, ACM, v. 54, n. 5, p. 49–53, 2011. Citado na página 36.
- COMPANY, B. A. S. L. *WT41 World's Longest Range Bluetooth Module*. 2017. BlueGiga WebSite. Disponível em: <<https://www.bluegiga.com/en-US/products/wt41-world-s-longest-range/>>. Acesso em: 21 de mai 2017. Citado na página 32.
- COULOURIS, G. et al. *Sistemas Distribuídos-: Conceitos e projeto*. [S.l.]: Bookman Editora, 2013. Citado 2 vezes nas páginas 43 e 56.

- CROW, B. P. et al. Ieee 802.11 wireless local area networks. *IEEE Communications magazine*, IEEE, v. 35, n. 9, p. 116–126, 1997. Citado na página 18.
- DJUKNIC, G. M.; RICHTON, R. E. Geolocation and assisted gps. *Computer*, IEEE, v. 34, n. 2, p. 123–125, 2001. Citado na página 33.
- ESPRESSIF. *ESP8266*. 2017. ESPRESSIF WebSite. Disponível em: <<https://espressif.com/en/products/hardware/esp8266ex/overview>>. Acesso em: 21 de mai 2017. Citado na página 40.
- FAHMI, P. A. et al. 3d-to-2d projection algorithm for remote control using smartphone: Enhancing smartphone capability for costless wireless audio visual consumer appliance control. In: IEEE. *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*. [S.l.], 2013. p. 1044–1049. Citado na página 32.
- FELICE, F. Análise do desempenho de enlaces ponto-a-ponto utilizando a faixa de frequência não licenciada de 2, 4ghz em tecnologia spread spectrum. 2005. Citado 2 vezes nas páginas 18 e 30.
- FERREIRA, A. B. d. H. *Novo dicionário da língua portuguesa*. [S.l.]: Nova Fronteira, 1986. Citado na página 35.
- GONZÁLEZ, V. et al. Experimental demonstration of the viability of ieee 802.11 b based inter-vehicle communications. In: ICST (INSTITUTE FOR COMPUTER SCIENCES, SOCIAL-INFORMATICS AND TELECOMMUNICATIONS ENGINEERING). *Proceedings of the 4th International Conference on Testbeds and research infrastructures for the development of networks & communities*. [S.l.], 2008. p. 1. Citado na página 20.
- GUO, M.; AMMAR, M. H.; ZEGURA, E. W. V3: A vehicle-to-vehicle live video streaming architecture. *Pervasive and Mobile Computing*, Elsevier, v. 1, n. 4, p. 404–424, 2005. Citado na página 29.
- IB, A. G. et al. A performance study of the 802.11 p standard for vehicular applications. In: IEEE. *Intelligent environments (IE), 2011 7th international conference on*. [S.l.], 2011. p. 165–170. Citado na página 28.
- JIANG, D.; DELGROSSI, L. Ieee 802.11 p: Towards an international standard for wireless access in vehicular environments. In: IEEE. *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*. [S.l.], 2008. p. 2036–2040. Citado na página 25.
- KAPLAN, E.; HEGARTY, C. *Understanding GPS: principles and applications*. [S.l.]: Artech house, 2005. 1068 p. Citado na página 33.
- KARNIK, A.; KUMAR, A. Performance analysis of the bluetooth physical layer. In: IEEE. *Personal Wireless Communications, 2000 IEEE International Conference on*. [S.l.], 2000. p. 70–74. Citado na página 31.
- LECHETA, R. *Google Android: Aprenda a criar aplicações para dispositivos móveis com o android sdk*. São Paulo: Novatec, 2016. 1068 p. (5ª Edição). ISBN 9788575224687. Citado 2 vezes nas páginas 36 e 40.

- LEE, J.-S.; SU, Y.-W.; SHEN, C.-C. A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. In: IEEE. *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE*. [S.l.], 2007. p. 46–51. Citado na página 31.
- LEE, K. C. et al. First experience with cartorrent in a real vehicular ad hoc network testbed. In: IEEE. *2007 Mobile Networking for Vehicular Environments*. [S.l.], 2007. p. 109–114. Citado na página 29.
- LEE, U. et al. Code torrent: content distribution using network coding in vanet. In: ACM. *Proceedings of the 1st international workshop on Decentralized resource sharing in mobile computing and networking*. [S.l.], 2006. p. 1–5. Citado na página 29.
- LEICK, A.; RAPOPORT, L.; TATARNIKOV, D. *GPS satellite surveying*. [S.l.]: John Wiley & Sons, 2015. Citado na página 33.
- MACEDO, R. et al. Uma avaliação experimental de desempenho do roteamento multicaminhos em redes veiculares. In: *Workshop de Gerência e Operações de Redes-SBRC*. [S.l.: s.n.], 2013. Citado na página 23.
- MARTELLI, F.; RENDA, M. E.; SANTI, P. Measuring iee 802.11 p performance for active safety applications in cooperative vehicular systems. In: IEEE. *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*. [S.l.], 2011. p. 1–5. Citado na página 20.
- MOBILE, V. Developer economics q3 2014: State of the developer nation. *Accessed*, v. 10, p. 14, 2014. Citado na página 36.
- NANDAN, A. et al. Co-operative downloading in vehicular ad-hoc wireless networks. In: IEEE. *Wireless On-demand Network Systems and Services, 2005. WONS 2005. Second Annual Conference on*. [S.l.], 2005. p. 32–41. Citado na página 29.
- OLIVEIRA, R. d. R. et al. Protocolo adaptativo de disseminação de dados para aplicações de segurança no trânsito em rodovias. 2013. Citado na página 34.
- OTHMAN, Z.; WAN, A. W.; ANUAR, A. Evaluating the performance of gps survey methods for landslide monitoring at hillside residential area: static vs rapid static. In: IEEE. *Signal Processing and its Applications (CSPA), 2011 IEEE 7th International Colloquium on*. [S.l.], 2011. p. 453–459. Citado na página 33.
- PANAYAPPAN, R. et al. Vanet-based approach for parking space availability. In: ACM. *Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*. [S.l.], 2007. p. 75–76. Citado 2 vezes nas páginas 17 e 29.
- PIRMEZ, L.; REZENDE, J. F. de. Mecanismos de escalonamento com qualidade de serviço em redes com tecnologia bluetooth. 2003. Citado na página 30.
- PLAY, G. *Google Play Store Apps*. 2017. Google Play apps. Disponível em: <<https://play.google.com/store/apps?hl=pt-br>>. Acesso em: 16 de ago 2016. Citado na página 35.
- PRINZ, V.; BROCCO, M.; WOERNDL, W. Peer-to-peer algorithms for c2c communication systems. In: ACM. *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia*. [S.l.], 2008. p. 376–381. Citado na página 29.

- PURI, P.; SINGH, M. P. A survey paper on routing in delay-tolerant networks. In: IEEE. *Information Systems and Computer Networks (ISCON), 2013 International Conference on*. [S.l.], 2013. p. 215–220. Citado na página 33.
- RAPPAPORT, T. S. *Comunicações sem fio: princípios e práticas*. [S.l.]: Pearson Prentice Hall, 2009. (2ª Edição). ISBN 9788576051985. Citado na página 31.
- SAÚDE, O. M. de. *Relatório global sobre o estado da segurança viária 2015*. 2015. Site oficial. Disponível em: <http://www.who.int/violence_injury_prevention/road_safety_status/2015/Summary_GSRRS2015_POR.pdf?ua=1>. Acesso em: 26 de set 2016. Citado na página 17.
- SIG, B. *Bluetooth 5: What it's all about*. 2017. Bluetooth Technology Website. Disponível em: <<https://www.bluetooth.com/specifications/bluetooth-core-specification/bluetooth5>>. Acesso em: 21 de mai 2017. Citado na página 32.
- SIG, B. *Bluetooth BR/EDR*. 2017. Bluetooth Technology Website. Disponível em: <<https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works/br-edr>>. Acesso em: 21 de mai 2017. Citado na página 32.
- SIG, B. *Bluetooth High Speed*. 2017. Bluetooth Technology Website. Disponível em: <<https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works/bluetooth-high-speed>>. Acesso em: 21 de mai 2017. Citado na página 32.
- SIG, B. *Bluetooth Low Energy*. 2017. Bluetooth Technology Website. Disponível em: <<https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works/low-energy>>. Acesso em: 21 de mai 2017. Citado na página 32.
- SIG, B. *How it Works*. 2017. Bluetooth Technology Website. Disponível em: <<https://www.bluetooth.com/what-is-bluetooth-technology/how-it-works>>. Acesso em: 21 de mai 2017. Citado 2 vezes nas páginas 30 e 31.
- SILVA, R. F. *Caracterização da mobilidade veicular e o seu impacto nas redes veiculares tolerantes a atrasos e desconexões*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2010. Citado 3 vezes nas páginas 17, 23 e 27.
- SOUSA, R. de; SOARES, A. Estimativa e sinalização de congestionamentos de trafego através de redes veiculares v2v. *XXXIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, Vitória*, 2015. Citado na página 21.
- SOUSA, R. S. de; SOARES, A. C. B. Estimativa e sinalização de congestionamentos de trafego através de redes veiculares v2v. *XXXIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, Vitória*, 2015. Citado 3 vezes nas páginas 17, 24 e 28.
- SOUZA, F. *Arduino Mega 2560*. 2017. Embarcados. Disponível em: <<https://www.embarcados.com.br/arduino-mega-2560/>>. Acesso em: 24 de abr 2017. Citado na página 43.
- SUKUVAARA, T. Field measurements of ieee 802.11 p based vehicular networking entity. In: IEEE. *Ubiquitous and Future Networks (ICUFN), 2012 Fourth International Conference on*. [S.l.], 2012. p. 135–139. Citado na página 20.

- TEIXEIRA, F. et al. Análise experimental de redes veiculares utilizando o padrão ieee 802.11 p. *Anais do V Simpósio Brasileiro de Computação Ubíqua e Pervasiva (SBCUP)*, p. 10, 2013. Citado na página 20.
- TRINDADE, I. *IPEA estima custo anual com acidentes no Brasil em 40 bilhões*. 2012. Congresso Internacional de Trânsito. Disponível em: <<http://www.congressodetransito.rs.gov.br/noticias/detalhe/20/IPEA-estima-custo-anual-com-acidentes-no-Brasil-em-R-40-bilhes>>. Acesso em: 19 de abr 2017. Citado na página 17.
- UZCÁTEGUI, R. A.; SUCRE, A. J. D.; ACOSTA-MARUM, G. Wave: A tutorial. *IEEE Communications magazine*, IEEE, v. 47, n. 5, 2009. Citado na página 26.
- VAHDAT, A.; BECKER, D. et al. Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke University, 2000. Citado 2 vezes nas páginas 20 e 34.
- XU, Q. et al. Vehicle-to-vehicle safety messaging in dsrc. In: ACM. *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*. [S.l.], 2004. p. 19–28. Citado na página 19.
- ZALDIVAR, J. et al. Providing accident detection in vehicular networks through obd-ii devices and android-based smartphones. In: IEEE. *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*. [S.l.], 2011. p. 813–819. Citado na página 19.
- ZHAO, J.; CAO, G. Vadd: Vehicle-assisted data delivery in vehicular ad hoc networks. *IEEE transactions on vehicular technology*, IEEE, v. 57, n. 3, p. 1910–1922, 2008. Citado na página 24.