

LAURO DE LACERDA CAETANO
ARTHUR ALBUQUERQUE ZOPELLARO SOARES

Redes Veiculares: Tendências e Estudo de Caso

Campos dos Goytacazes

2016

LAURO DE LACERDA CAETANO
ARTHUR ALBUQUERQUE ZOPELLARO SOARES

Redes Veiculares: Tendências e Estudo de Caso

Trabalho de conclusão de curso apresentado
ao Instituto Federal de Educação, Ciência e
Tecnologia Fluminense como requisito par-
cial para conclusão do curso de Bacharelado
em Sistemas de Informação.

Instituto Federal Fluminense – IFF
Campus Campos Centro
Sistemas de Informação

Orientador: MSc. VINÍCIUS BARCELOS

Campos dos Goytacazes
2016

LAURO DE LACERDA CAETANO
ARTHUR ALBUQUERQUE ZOPELLARO SOARES
Redes Veiculares: Tendências e Estudo de Caso/ LAURO DE LACERDA
CAETANO
ARTHUR ALBUQUERQUE ZOPELLARO SOARES. – Campos dos Goytacazes,
2016-
156 p. : il. (algumas color.) ; 30 cm.

Orientador: MSc. VINÍCIUS BARCELOS

Trabalho de Conclusão de Curso – Instituto Federal Fluminense – IFF
Campus Campos Centro
Sistemas de Informação, 2016.

1. Redes Veiculares. 2. WAVE. 3. VEINS. I. MSc. Vinícius Barcelos. II.
Instituto Federal Fluminense. III. Campus Campos Centro. IV. Título

LAURO DE LACERDA CAETANO
ARTHUR ALBUQUERQUE ZOPELLARO SOARES

Redes Veiculares: Tendências e Estudo de Caso

Trabalho de conclusão de curso apresentado
ao Instituto Federal de Educação, Ciência e
Tecnologia Fluminense como requisito par-
cial para conclusão do curso de Bacharelado
em Sistemas de Informação.

Trabalho aprovado. Campos dos Goytacazes,

MSc. VINÍCIUS BARCELOS
INSTITUTO FEDERAL FLUMINENSE - IFF
Orientador

D.Sc.
INSTITUTO FEDERAL FLUMINENSE - IFF

M.Sc. BANCA 2
INSTITUTO FEDERAL FLUMINENSE - IFF

M.Sc. BANCA 2
INSTITUTO FEDERAL FLUMINENSE - IFF

Campos dos Goytacazes
2016

Agradecimentos

*"Nothing is too wonderful to be true,
if it be consistent with the laws of nature."*
(Michael Faraday)

Resumo

Este trabalho descreve o desenvolvimento da tecnologia de redes veiculares, sua padronização e suas ferramentas. Em especial, a norma IEEE 1609 é investigada em maior detalhe. São citados diversos exemplos de aplicações possíveis para a tecnologia veicular. Encontram-se analisados diversos softwares de simulação que, quando integrados, permitem que os veículos se conectem em rede. A tecnologia é simulada e analisada através de um estudo de caso em um cenário na cidade do Rio de Janeiro. Tutoriais de instalação e de criação de novos cenários de simulação estão adicionados neste trabalho.

Palavras-chaves: Redes Veiculares, WAVE, VEINS

Abstract

This work describes the development of vehicular networking technology, its standardization and tools. Specifically, the IEEE 1609 standard is investigated in detail. Several examples of vehicular technology applications are given. A group of simulation tools is analyzed and compared. When some of them are integrated, they allow vehicles to be connected in networks. The technology is simulated and scrutinized on a case study in a scenario in Rio de Janeiro city. Installation and implementation guides that help creating new scenarios are attached to this work.

Key-words: Vehicular Networks, WAVE, VEINS

Listas de ilustrações

Figura 1 – Modelo OSI	24
Figura 2 – Modelo OSI e TCP/IP	25
Figura 3 – Encapsulamento de dados	27
Figura 4 – Desencapsulamento de dados	28
Figura 5 – <i>Unicast</i>	29
Figura 6 – Broadcast	30
Figura 7 – <i>Multicast</i>	31
Figura 8 – Modo Infra-estruturado	34
Figura 9 – Modo de Operação Ad-Hoc	34
Figura 10 – Arquitetura IEEE 802.11 - IBSS	35
Figura 11 – Arquitetura IEEE802.11 - BSS	35
Figura 12 – Protocolos de Roteamento em MANETs	38
Figura 13 – VANETs	41
Figura 14 – OBU produzida pela Cohda Wireless	42
Figura 15 – RSU produzida pela Cohda Wireless	43
Figura 16 – Diversas OBUs e RSUs	44
Figura 17 – Arquitetura WAVE - Planos	45
Figura 18 – Segurança WAVE Detalhada	48
Figura 19 – <i>Service Access Points</i>	49
Figura 20 – Espectro Eletromagnético	50
Figura 21 – Espectro WAVE FCC EUA	51
Figura 22 – Tipos de Canais	53
Figura 23 – Exemplo de MAC Comum e MAC WAVE	54
Figura 24 – Exemplos de Acesso ao Canal	56
Figura 25 – Intervalo de Sincronismo e de Guarda	57
Figura 26 – MAC(Multi-Canal)	58
Figura 27 – Cabeçalho LLC	61
Figura 28 – Escopo do WSMP nesta seção	63
Figura 29 – Composição de uma WSM	64
Figura 30 – Cabeçalho de Rede WSMP Padrão(Subtipo = 0)	65
Figura 31 – Cabeçalho de Transporte WSMP Padrão	65
Figura 32 – Diferentes formatos do Cabeçalho WSMP-T de acordo com TPID	66
Figura 33 – Modelos de uma WSA	67
Figura 34 – Cabeçalho de WSA	68
Figura 35 – Transmissão e Recepção de uma WSM	69
Figura 36 – Atribuição de IPv6 por uma unidade de acostamento	71

Figura 37 – Aplicação de pedágio	74
Figura 38 – Interface OMNET++	81
Figura 39 – Interface TkEnv	82
Figura 40 – Interface QtEnv	82
Figura 41 – Module Logger	83
Figura 42 – Message Logger	83
Figura 43 – Interface SUMO	85
Figura 44 – Diagrama Simplificado de Classes do VEINS	92
Figura 45 – Diagrama de Classes para o Estudo de Caso	93
Figura 46 – Cenários dos Experimentos 1 e 2 (escala reduzida)	97
Figura 47 – Computador de Bordo solicitando abertura de passagem (figura conceptual)	98
Figura 48 – Veículo de Resgate envia mensagem de broadcast	99
Figura 49 – Apenas os veículos da pista inferior mudam de faixa após mensagem	99
Figura 50 – Veículo azul em menor velocidade porém não é seguro mudar de faixa	100
Figura 51 – Após alguns instantes a mudança de faixa é realizada	100
Figura 52 – Veículo vermelho é influenciado pela mensagem de emergência	101
Figura 53 – Veículo vermelho muda de faixa e realiza a ultrapassagem	101
Figura 54 – Raio de alcance das RSUs	102
Figura 55 – Experimentos 1 e 2 - Comparação do tempo de recebimento das mensagens	103
Figura 56 – Trajetória do veículo de resgate e RSUs	104
Figura 57 – Disposição das RSUs no Experimento 3	105
Figura 58 – Fluxo dos Carros e obstáculos em vermelho no Experimento 3	106
Figura 59 – Gráfico de Medição Base - <i>Outliers</i> nos círculos vermelhos	107
Figura 60 – Gráfico de Aplicação de Emergência ativada - <i>Outliers</i> nos círculos vermelhos	108
Figura 61 – Gráfico de Aplicação de Emergência ativada em cenário com semáforos abertos - <i>Outliers</i> nos círculos vermelhos	109
Figura 62 – MIB IPv6	120
Figura 63 – MIBs - Interfaces entre camadas	121
Figura 64 – Estrutura do campo Informações Extensíveis do Elemento WAVE	124
Figura 65 – Estrutura de uma Informações do Elemento WAVE	124
Figura 66 – Configuração bem sucedida do OMNET++ no Linux	130
Figura 67 – Configuração mal sucedida do OMNET++ no Linux	130
Figura 68 – Software mingwenv em execução no Windows	132
Figura 69 – Configuração do Terminal Virtual no Windows	133
Figura 70 – Terminal Virtual no Windows	133
Figura 71 – Configuração bem sucedida do OMNET++ no Windows	134

Figura 72 – Instalação bem sucedida do OMNET++ no Windows	135
Figura 73 – Abrindo uma Simulação no SUMO	136
Figura 74 – Inicializando uma Simulação no SUMO	136
Figura 75 – Visualizando uma Simulação no SUMO	137
Figura 76 – Executando o OMNET++	138
Figura 77 – Tela de Boas vindas do OMNET++	138
Figura 78 – Importação de Exemplos	139
Figura 79 – Importação do VEINS	140
Figura 80 – VEINS importado no OMNET++	140
Figura 81 – Construindo o VEINS no OMNET++	141
Figura 82 – VEINS Construído com Sucesso	141
Figura 83 – Executando o <i>daemon</i> no Linux	142
Figura 84 – Executando o <i>daemon</i> no Windows	143
Figura 85 – Inicializando a Simulação Teste	144
Figura 86 – Configuração de Execução Criada	144
Figura 87 – Executando a Simulação Teste	145
Figura 88 – Simulação Teste em Execução (SUMO e Tkenv)	145
Figura 89 – Troca de Mensagens entre os Veículos	146
Figura 90 – Pesquisando um Endereço no OpenStreetMap	147
Figura 91 – Habilitando Exportação de Mapas no OpenStreetMap	148
Figura 92 – Habilitando Seleção de Área no OpenStreetMap	149
Figura 93 – Selezionando Área para Exportar no OpenStreetMap	149
Figura 94 – Exportando Mapa no OpenStreetMap	150
Figura 95 – Alterando o <i>Delay</i>	152
Figura 96 – Configurando NED para a nova Simulação	155
Figura 97 – Nova Simulação no <i>Project Explorer</i>	155

Lista de tabelas

Tabela 1 – Padrões IEEE 802.11	33
Tabela 2 – Família de Padrões WAVE	46
Tabela 3 – Entidades de Gerenciamento	47
Tabela 4 – Dados da aplicação de pedágio	75
Tabela 5 – Comparação entre softwares de simulação de redes	79
Tabela 6 – Modelos de tráfego por granularidade	84
Tabela 7 – Módulos do VEINS	88
Tabela 8 – Arquivos do OMNET++	89
Tabela 9 – Descrição das Classes do VEINS	92
Tabela 10 – Descrição das Classes desenvolvidas	94
Tabela 11 – Dispositivos Car2X no cenário dos experimentos 1 e 2	97
Tabela 12 – Resultados Experimento 1	102
Tabela 13 – Resultados Experimento 2	103
Tabela 14 – Dispositivos Car2X no cenário dos experimento 3	105
Tabela 15 – Resultados do Experimento 3 - Medição base	107
Tabela 16 – Resultados do Experimento 3 - Medição com aplicação ativada	108
Tabela 17 – Resultados - Medição com aplicação ativada e semáforos abertos	109
Tabela 18 – Resultados Experimentos do Estudo de Caso	110
Tabela 19 – PSID - Alocação de Identificadores	123
Tabela 20 – Descrição dos diversos Identificadores do Elemento WAVE	125
Tabela 21 – Bibliotecas necessárias para Sistemas Unix	127

Lista de abreviaturas e siglas

ANATEL	Agência Nacional de Telecomunicações
ASN.1	Abstract Syntax Notation One
BSM	Basic Safety Message
BSS	Basic Service Set
BSSID	BSS Identification
CCH	Control Channel
DNS	Domain Name System
DS	Distribution System
DSRC	Dedicated Short Range Communications
EDCA	Enhanced Distributed Channel Access
ETSI	European Telecommunications Standards Institute
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronics Engineers
IBSS	Independent BSS
IoV	Internet of Vehicles
ITS	Intelligent Transportation Systems
LAN	Local Area Network
LLC	Logical Link Control
MAC	Media Access Control
MANET	Mobile Ad Hoc Network
MIMO	Multiple Input Multiple Output
MLME	MAC subLayer Management Entity
MLMEX	MAC subLayer Management Entity Extension

NED	Network Description
OFDM	Orthogonal Frequency Division Multiplexing
OSI	Open Systems Interconnection
OCB	On the Context of BSS
OBU	On Board Unit
PHY	Physical Layer
PLME	Physical Layer Management Entity
PSC	Provider Service Context
PSID	Provider Service Identifier
RSA	Roadside Alert
RSU	Roadside Unit
SAE	Society of Automotive Engineers
SCH	Service Channel
SDEE	Secure Data Exchange Entity
SSME	Security Services Management Entity
SUMO	Simulator of Urban Mobility
TraCI	Traffic Command Interface
TPID	Transport Protocol Identification
UTC	Universal Time Coordinated
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
VANET	Vehicle Ad Hoc Network
VEINS	Vehicles in Network Simulation
WAVE	Wireless Access in Vehicular Enviroments
WME	WAVE Management Entity
WRA	WAVE Routing Advertisement

WSA	WAVE Service Advertisement
WSM	WAVE Short Message
WSMP	WAVE Short Message Protocol
WSMP-N	WSMP Network
WSMP-T	WSMP Transport

Sumário

1	INTRODUÇÃO	19
1.1	Justificativa	21
1.2	Objetivos	21
1.3	Metodologia	22
2	INTRODUÇÃO À REDES SEM FIO	23
2.1	Modelos de Referência	23
2.1.1	Modelo OSI	23
2.1.2	Modelo TCP/IP	25
2.2	Encapsulamento de dados	26
2.3	<i>Unicast, Multicast e Broadcast</i>	29
2.4	Redes LAN	31
2.5	Família de padrões IEEE 802	32
2.6	Família de padrões IEEE 802.11	32
2.6.1	Modos de operação	33
2.6.2	Arquitetura IEEE 802.11	34
2.6.3	Serviços IEEE 802.11	36
2.7	Redes Ad-hoc Móveis	37
3	REDES DE COMUNICAÇÃO VEICULARES	39
3.1	História	40
3.2	VANETs	41
3.3	Dispositivos Car2X: Componentes e Tipos de Comunicação	42
3.4	Família de padrões WAVE	44
3.5	Plano de Gerenciamento	46
3.5.1	Camada de Gerenciamento	47
3.5.2	Camada de Segurança	47
3.5.3	Interfaces	49
3.6	Camada Física	50
3.6.1	Tipos de Canais	52
3.7	Subcamada MAC	54
3.7.1	Operações multi-canal	55
3.7.2	Serviços do Plano de Dados	55
3.7.3	Serviços do Plano de Gerenciamento	58
3.8	Subcamada LLC	60
3.9	Camadas de Rede e Transporte	62

3.9.1	Protocolo WSMP	62
3.9.1.1	Mensagem Curta WAVE	64
3.9.1.2	Mensagem de Anúncio WAVE	67
3.9.1.3	Transmissão e recepção de mensagens curtas WAVE	68
3.9.2	Protocolo IPv6	70
3.10	Camada de Aplicação	71
3.10.1	Aplicações de Segurança e Informação	71
3.10.2	Coleta eletrônica de Taxas	74
4	ABORDAGEM AVALIATIVAS	76
4.1	Softwares de Simulação de Redes	77
4.1.1	OMNET++	80
4.2	Software de Simulação de Tráfego	84
4.2.1	<i>Simulator of Urban Mobility</i>	85
4.3	Middleware de Integração	86
4.3.1	<i>Vehicles in Network Simulation</i>	86
5	ESPECIFICAÇÕES DO SISTEMA E ESTUDO DE CASO	91
5.1	Especificações do Sistema	91
5.1.1	Aplicação de Emergência	92
5.2	Estudo de Caso	96
5.2.1	Descrição dos Experimentos	96
5.2.2	Experimentos 1 e 2: Descrição do Cenário	96
5.2.2.1	Análise e Resultados: Experimento 1	98
5.2.2.2	Análise e Resultados: Experimento 2	102
5.2.3	Experimento 3: Aplicação de Emergência na região de Ipanema-Copacabana	103
5.2.3.1	Análise e Resultados: Experimento 3	106
5.3	Conclusão: Estudo de Caso	109
	CONCLUSÃO	112
	TRABALHOS FUTUROS	113
	REFERÊNCIAS	114
	ANEXOS	118
	ANEXO A – ABSTRACT SYNTAX NOTATION ONE	119
	ANEXO B – MANAGEMENT INFORMATION BASE	120

ANEXO C – ALOCAÇÃO DE PSIDS	122
ANEXO D – INFORMAÇÕES DO ELEMENTO WAVE	124
ANEXO E – PREPARANDO O AMBIENTE PARA SIMULAÇÕES VEICULARES NO VEINS	126
E.1 Sistemas Linux	127
E.1.1 Instalação do SUMO	128
E.1.2 Instalação do OMNET++	129
E.2 Sistemas Windows	131
E.2.1 Instalação do SUMO	131
E.2.2 Instalação do OMNET++	132
E.3 Testar funcionamento do SUMO	135
E.4 Configuração do OMNET++	137
E.5 Importação do VEINS	139
E.5.1 Testar integração dos <i>Frameworks</i>	142
ANEXO F – DESENVOLVENDO SIMULAÇÕES	147
F.1 Exportando mapas através do OpenStreetMap	147
F.2 Criando polígonos para o mapa	150
F.3 Gerando rotas aleatórias	151
F.4 Preparando a simulação	152
F.5 Simulando a Rede Veicular	153
ANEXO G – TABELAS E CÓDIGO FONTE	156

1 Introdução

Com objetivo de promover integração mundial e contribuir para o crescimento da economia, cooperando no deslocamento de pessoas e mercadorias, os meios de transporte são objetos fundamentais na atual sociedade. Estes determinam aquilo que é chamado de “compressão do tempo e do espaço”, onde as distâncias do mundo se encurtam, ou seja, podem ser mais facilmente percorridas, reduzindo gastos e tempo de trajeto (Harvey, 1989).

Aliados aos meios de transporte, os sistemas de comunicação cooperam para o aumento da segurança e da comodidade nos percursos. Em ambiente marítimo, itens como rádios auxiliam na conversação entre embarcações e bases marítimas. Transponders e sonares ajudam na transmissão de coordenadas e no descobrimento de embarcações próximas.

Nos transportes aéreos, o mesmo transponder passou a ser utilizado na aviação comercial, tornando-se item obrigatório. Este coopera para a comunicação de aviões com torres de comando além de ser parte importante dos sistemas anticolisão. Através da troca de informação entre estes sistemas, é possível que aviões projetem informações de outras aeronaves em um raio considerável, como localização, rumo, altitude e velocidade.

Os transportes terrestres, contudo, não evoluíram na mesma velocidade em que os veículos marítimos e aéreos no que diz respeito aos sistemas de comunicação. Dadas as claras **externalidades**¹ negativas causadas pelos veículos nas cidades como trânsito, poluição e acidentes, vê-se como necessário o pronto desenvolvimento de ferramentas de comunicação que auxiliem na mitigação destas externalidades.

Dotados de maior conscientização sobre os fatos, os governos têm examinado o papel do veículo na cadeia de valor da indústria automobilística, buscando tratar as externalidades citadas. Além disso, a comunidade científica tem trabalhado no desenvolvimento de mecanismos que visam melhorar a qualidade de vida dos seres humanos em ambiente veicular.

A Internet dos Veículos apresenta-se como um conceito que expande o grau de ubiquidade da tecnologia da informação para o meio veicular. Como tecnologia, a internet dos veículos tem por objetivo a aplicação de recursos para solução ou mitigação de problemas, além servir de ferramenta para a geração de novos conhecimentos. E como tecnologia da informação, esta permite o tratamento e a difusão de dados. Portanto, a eventual consolidação deste conceito no mercado possibilitaria o surgimento de um novo paradigma de Internet, abrindo um amplo horizonte para a ciência com o surgimento de

¹ efeitos sociais, econômicos e ambientais indiretamente causados pela venda de um produto ou serviço

novas ferramentas de comunicação, entretenimento e segurança.

Vista como subclasse da Internet das Coisas, a Internet dos Veículos é uma rede que também integra dispositivos incorporados de sensores (veículos, itens de sinalização, unidades de acostamento), software e rede de dados, possibilitando que veículos e mais objetos coletem e troquem informações. A partir do processamento das informações, os veículos podem tornar-se mais inteligentes, sendo capazes de auxiliar na tomada de decisões até de maneira parcial ou totalmente autônomas, visando a segurança e bem estar dos passageiros e pedestres.

As redes de comunicação veiculares, parte fundamental para incremento da inteligência no meio veicular, continuam em plena evolução. O estudo do funcionamento destas tem mais de uma década, sendo iniciado pelas VANETs (*Vehicular Ad Hoc Networks* ou Redes Ad Hoc Veiculares), passando por diversos protocolos de teste, até chegar ao estabelecimento de famílias de padrões. Duas destas se destacam, a primeira família chamada de Acesso sem fio em Ambientes Veiculares (WAVE) contém um conjunto de padrões criados pelo Instituto de Engenheiros Elétricos e Eletrônicos(IEEE) que foram submetidos a diversas baterias de testes e provas de conceito. Na Europa, a família de padrões para Sistemas de Transportes Inteligentes do Instituto de Padrões de Telecomunicações Europeu(ETSI) se iguala em notoriedade na comunidade científica.

Segundo expectativas de mercado, 75% dos carros do mundo estarão conectados em 2020, e o crescimento da Internet dos Veículos trará em torno de US\$2,94 bilhões em receitas². Com um amplo apoio do Departamento de Transporte dos Estados Unidos, a comunicação entre veículos está perto de se tornar obrigatória para automóveis em circulação neste país³. Na Europa, a implantação da comunicação veicular também está acelerada. Foi criado um consórcio chamado Car2Car a partir da iniciativa de fabricantes de veículos europeus. Este consórcio sem fins lucrativos tem por objetivo reforçar a eficiência e segurança no tráfego rodoviário, utilizando de comunicações entre veículos.

As redes veiculares apresentam-se como área promissora para a ciência e para o mercado de trabalho. É possível que nos próximos anos apareçam *startups* focadas no desenvolvimento de aplicações voltadas para entretenimento e segurança veicular. Além disso, empresas de tecnologia 3G e 4G podem se aproveitar de uma ampliação do alcance das comunicações interveiculares para oferecer planos de dados, transformando os veículos numa extensão da atual Internet. Ademais, com o advento dos veículos autônomos servindo-se da imprescindível existência de redes de comunicação, podemos visionar o aumento da segurança dos transportes e do bem estar de condutores e passageiros.

² <http://press.trendforce.com/node/view/1880.html> acesso em 13 de maio 2016

³ <https://www.transportation.gov/fastlane/dialogue-industry-conversation-between-cars> acesso em 14 de maio 2016

1.1 Justificativa

O atual mercado conta com diversas ferramentas para irradiação de informações no trânsito. Assim como painéis digitais em vias de alta densidade veicular cooperam para melhor fluidez do trânsito, redes sociais como o Waze alertam eventuais acidentes no caminho de condutores.

Entretanto, foi percebido que os atuais meios de comunicação de tráfego rodoviário ainda são limitados no que diz respeito à propagação rápida de mensagens de emergência no trânsito. Isto ocorre pois a maioria das ferramentas dependem do processamento de seus servidores antes da entrega da informação.

As Redes Veiculares, por usarem comunicação ponto-a-ponto, propagam a informação com menor atraso e por isso podem ser usadas para enviar informações de situações em tempo real a motoristas, ajudando na tomada de decisão dos condutores e prevenindo possíveis acidentes.

Equipes de resgate são constantemente acionadas para o socorro de pessoas e necessitam de tráfego livre no trajeto até o socorrido. Quando há trânsito no perímetro urbano ou em rodovias, as equipes de resgate têm seu trabalho atrapalhado por obstruções na passagem. Mesmo com giroscópio e sirene emitindo sinais visual e sonoro, o atraso no atendimento pode ser gerado por um efeito estocástico de obstruções. Como consequência do atraso até o local de atendimento, a demora para socorro de vítimas pode ser fatal.

Há, portanto, que se testar a utilização de métodos mais eficientes de propagação de mensagens utilizando-se de Redes Veiculares. Antes de entrar em produção, é preciso utilizar métodos de avaliação em caráter de simulação a fim de testar a transmissão destas mensagens de emergência entre unidades de acostamento e veículos. Seria possível antecipar situações de prováveis obstruções enviando mensagens aos carros das redondezas, acelerando a liberação de vias para um resgate com menor atraso.

1.2 Objetivos

Neste trabalho pretende-se desenvolver um estudo de caso em ambiente de simulação de microtráfego urbano empregando redes veiculares para a propagação em broadcast de mensagens oriundas de veículos de segurança pública.

Também é objetivo contribuir com uma ampla revisão bibliográfica para análise da família padrões IEEE 1609 WAVE, especialmente os já homologados pelo IEEE. Estes padrões formarão uma consistente base teórica para o estudo de caso.

Além disso, este trabalho tem finalidade de orientar através de tutoriais:

- a instalação de um conjunto de softwares acoplados para a simulação de redes veiculares.

ulares;

- a criação de novos cenários e experimentos de simulação utilizando esta tecnologia.

1.3 Metodologia

Como base teoria para um estudo de caso, serão discorridos conceitos de redes de comunicação e de redes veiculares, que foram padronizados pelo IEEE e outras organizações, através das normas IEEE 802.11p, IEEE 1609 WAVE e SAE J2735.

Serão analisados simuladores de tráfego urbano e simuladores de redes a fim de encontrar um grupo que melhor trabalhe com simulações de redes veiculares.

Após definição deste grupo, será desenvolvido um estudo de caso em redes veiculares analisando os efeitos de uma aplicação, criada pelos autores deste trabalho, que transmite mensagens de emergência no trânsito.

Como fruto do desenvolvimento do estudo de caso serão criados tutoriais para facilitar a instalação e a criação de novos estudos que investiguem as redes veiculares.

2 Introdução à Redes sem fio

Este capítulo apresenta as bases teóricas no estudo de redes de comunicação, seus principais padrões e características. Também serão estabelecidos conceitos iniciais de comunicação sem fio e suas normas, que servirão de base para um melhor entendimento dos padrões de redes veiculares que serão posteriormente discutidos neste trabalho.

2.1 Modelos de Referência

O estudo de redes de comunicação inicia no entendimento das duas mais relevantes arquiteturas de rede, os modelos de referência OSI e TCP/IP.

2.1.1 Modelo OSI

No começo dos anos 80, diversas empresas perceberam que deveriam sair do modelo proprietário dos sistemas de rede para resolver problemas de incompatibilidade e falta de comunicação. O modelo de referência OSI foi o primeiro modelo criado para redes de comunicação e teve, à época, o objetivo de ajudar estas organizações a criar redes que pudessem ser compatíveis e interoperáveis umas com as outras.

O modelo OSI foi liberado em 1984 contendo um conjunto de padrões que garantiram maior compatibilidade e operabilidade entre os vários tipos de tecnologias de rede produzidos por empresas ao redor do mundo. Apesar de outros modelos terem existido, o modelo de referência OSI é considerado a melhor ferramenta disponível para o envio e recebimento de dados em uma rede.

O Modelo OSI foi dividido em 7 níveis que se inter-relacionam. Foi feito desta maneira para facilitar o entendimento de como a informação viaja através de uma rede. Cada nível ilustra uma função particular da rede. Uma camada conta com a próxima camada inferior para realizar funções mais primitivas e ocultar os detalhes destas funções. Para exemplificar, o modelo OSI descreve como os dados trafegam de uma aplicação através de um meio de rede até uma aplicação localizada em outro computador, mesmo que o emissor e o receptor estejam utilizando meios de rede diferentes. A [Figura 1](#) apresenta a estrutura do modelo, composto de sete camadas ordenadas.



Figura 1 – Modelo OSI

Fonte: Autor

Cada camada OSI contém um conjunto de funções executadas por programas para fazer os dados viajarem de uma origem até o destino de uma rede.

A camada de aplicação é a mais próxima do usuário e provê serviços de rede para as aplicações. Esta, diferente dos outros níveis do modelo, não oferece serviços para outras camadas OSI superiores. Entretanto, realiza ajustes em procedimentos de recuperação de erros e no controle da integridade dos dados.

A camada de apresentação garante que os dados da camada de aplicação de um sistema são legíveis pela camada de aplicação de outro sistema. Se necessário, a camada de apresentação traduz e estrutura múltiplos formatos de dados, usando um formato comum.

Além de reportar problemas das camadas superiores, a camada de sessão fornece a estrutura de controle para comunicação entre as aplicações. Esta camada estabelece, gerencia e termina as conexões entre as aplicações.

A camada de transporte segmenta os dados do sistema emissor e reúne os dados em um fluxo único no sistema recipiente. Isto possibilita a transferência de dados confiável e transparente entre as extremidades, oferecendo recuperação de erro e controle de fluxo de ponta a ponta.

A camada de rede oferece conectividade e seleção de caminho entre dois sistemas que podem estar em redes separadas geograficamente. Nela há o roteamento de pacotes e é a responsável por estabelecer, manter e terminar as conexões.

A camada de enlace define como os dados são formatados para transmissão e como

o acesso à rede é controlado. Envia *frames* para sincronismo, controle de erro e controle de fluxo, se necessário.

A camada física define as especificações elétricas, mecânicas, procedurais e funcionais para ativação, manutenção e desativação de uma conexão física entre sistemas. A transmissão dos dados é binária. Características como níveis de voltagem, sincronismo das variações de tensão, taxas de dados físicos, distâncias máximas de transmissão, conectores físicos e outros atributos similares são definidos nas especificações da camada física.

2.1.2 Modelo TCP/IP

Embora o modelo de referência OSI seja reconhecido mundialmente, há um outro padrão que tem uma visão histórica e sobremaneira técnica da Internet: a pilha de protocolos TCP/IP.

A pilha de protocolos TCP/IP tem quatro camadas: Aplicação, Transporte, Internet e Acesso à Rede. Apesar de algumas camadas do modelo TCP/IP serem homônimas de algumas do modelo OSI, estas têm diferentes funções em cada modelo. A Figura 2 compara as camadas do modelo OSI e TCP/IP.

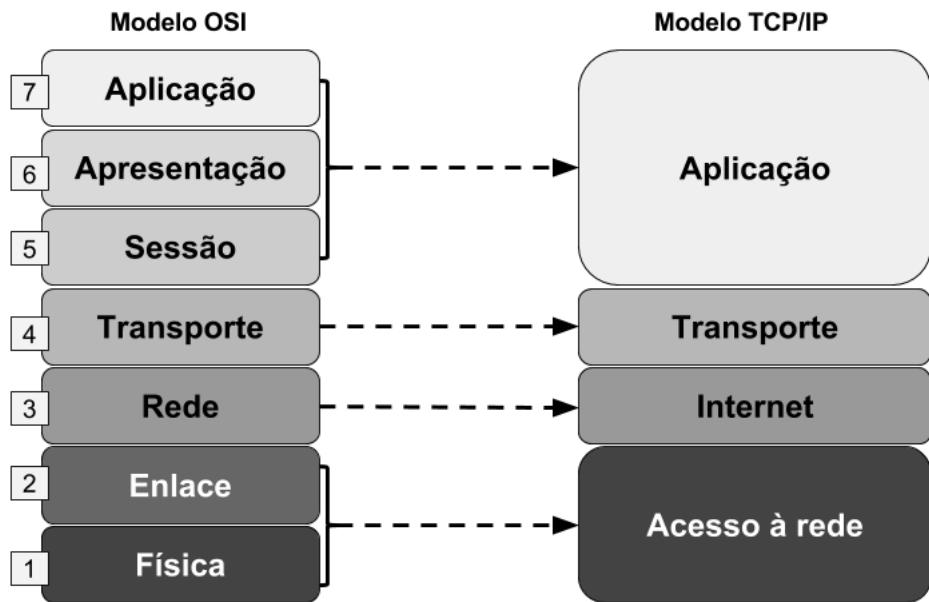


Figura 2 – Modelo OSI e TCP/IP

Fonte: Autor

A camada de aplicação controla protocolos de alto nível, incluindo questões de representação, codificação e controle de diálogo. O modelo TCP/IP combina todas as

questões relacionadas às aplicações em um único nível e garante que esses dados sejam devidamente empacotados para o próximo nível. Há diversos protocolos de aplicação, como FTP e o HTTP.

A camada de transporte lida com questões de Qualidade de Serviço (*Quality of Service* ou QoS) como confiabilidade, controle de fluxo e correção de erros. Um de seus protocolos é o TCP (*Transmission Control Protocol* ou Protocolo de Controle de Transmissão), que oferece comunicações de rede confiáveis.

A camada de Internet tem a responsabilidade de enviar pacotes através de redes. Para interligação dessas redes, é necessário que o envio de dados a partir de uma rede de origem chegue até a rede de destino. Este processo é chamado de roteamento ou encaminhamento.

A camada de acesso à rede inclui Protocolos de LAN e WAN e todos os detalhes da camada Física e de Enlace do modelo OSI.

2.2 Encapsulamento de dados

Todas as comunicações em uma rede iniciam em uma origem e são enviadas a um destino. Esta seção explica como o processo de transmissão de dados funciona em redes.

O conjunto de informações enviado em uma rede é chamado de quadro ou pacotes de dados. Se um computador deseja enviar dados para outro computador, os dados devem primeiro ser empacotados em um processo chamado encapsulamento. O encapsulamento “embrulha” os dados com informações necessárias do protocolo antes do envio do pacote na rede. Na medida em que os dados se movem da camada de aplicação até a camada física do modelo OSI, cada camada adiciona um cabeçalho (e um rodapé se for apropriado) aos dados antes da passagem para camada de baixo.

Os cabeçalhos contém informações de controle, garantindo que haja uma entrega apropriada dos dados e que o destinatário possa interpretar corretamente as informações.

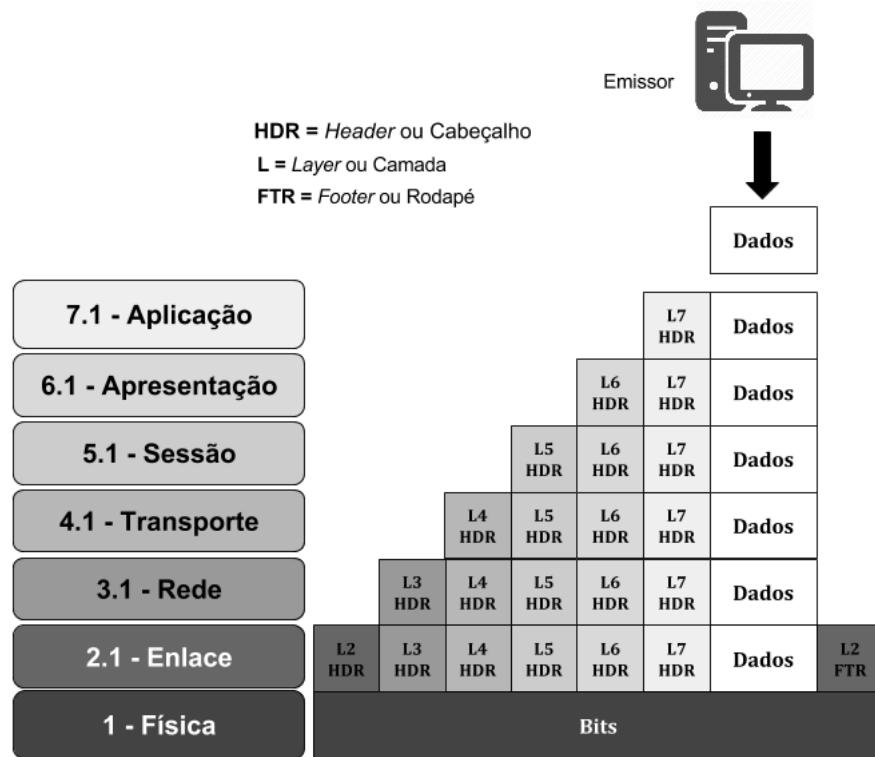


Figura 3 – Encapsulamento de dados

Fonte: Autor

Tomando como base o modelo OSI, o encapsulamento ocorre na seguinte ordem (Figura 3):

1. Os dados são enviados de uma aplicação para a camada de aplicação.
2. A camada de aplicação adiciona o seu cabeçalho aos dados. O pacote resultante é passado para a camada de apresentação.
3. A camada de apresentação adiciona o seu cabeçalho aos dados. O pacote resultante é passado para a camada de sessão.
4. A camada de sessão adiciona o seu cabeçalho aos dados. O pacote resultante é passado para a camada de transporte.
5. A camada de transporte adiciona o seu cabeçalho aos dados. O pacote resultante é então passado para a camada de rede.
6. A camada de rede adiciona o seu cabeçalho aos dados. O pacote resultante é então passado para a camada de enlace.
7. A camada de enlace adiciona o seu cabeçalho e também o rodapé (chamado de *footer* em inglês) aos dados. Este rodapé contém a checagem da sequência de quadros (FCS)

ou *Frame Check Sequence*), que é usada pelo destinatário para verificar se há algum erro nos dados. O pacote resultante é então passado para a camada física.

8. A camada física então transmite os bits para o meio de rede, que pode ser cabeados (fibra ótica, par trançado ou coaxial) ou sem fio (rádio, celular, infravermelho, *Bluetooth*).

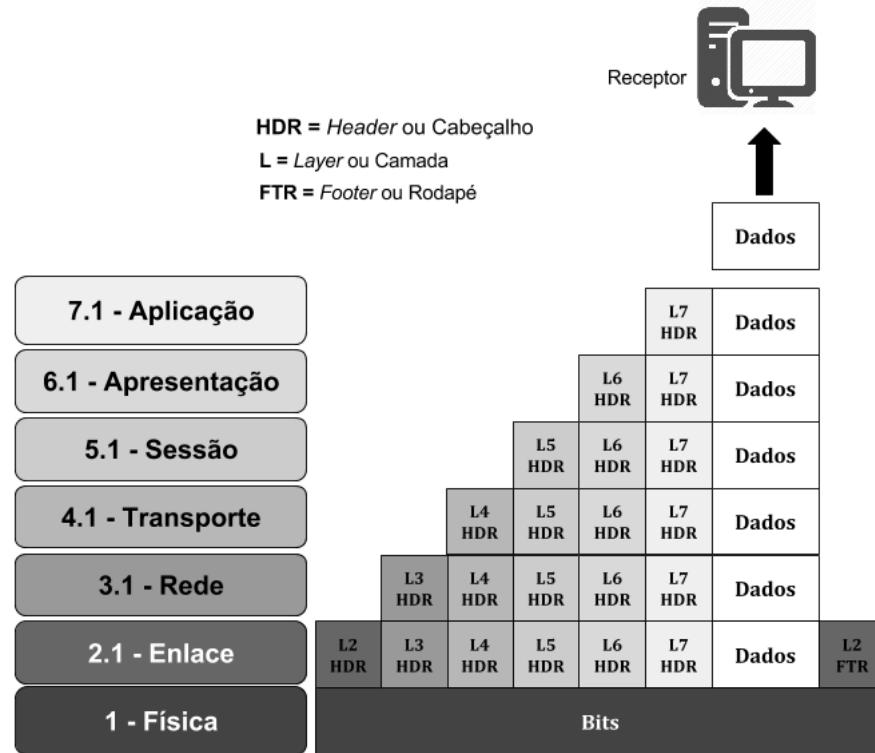


Figura 4 – Desencapsulamento de dados

Fonte: Autor

No desencapsulamento, vemos o processo acontecer de maneira invertida ([Figura 4](#)).

1. O dispositivo destinatário recebe a sequência de bits através do meio de rede e sua camada física entrega os bits para a camada de enlace.
2. A camada de enlace executa as seguintes ações:
 - Verifica o rodapé contendo o FCS para a detecção de erros.
 - Se houver erros nos dados, o pacote pode ser descartado, e a camada de enlace pode solicitar a retransmissão dos dados.
 - Se não houver erros, a camada de enlace lê e interpreta as informações de controle no cabeçalho da camada de enlace.

- Retira o cabeçalho da camada de enlace e o rodapé, e então passa o restante dos dados até a camada de rede baseando-se nas informações de controle do cabeçalho.
3. Das camadas de rede até apresentação os cabeçalhos são retirados e o restante dos dados entregue à camada de aplicação.
 4. A camada de aplicação recebe e interpreta os dados.

2.3 *Unicast, Multicast e Broadcast*

Existem três diferentes formas de compartilhamento de mensagens, chamados de *Unicast*, *Multicast* e *Broadcast*. Estes descrevem técnicas específicas para o envio da informação de acordo com cada possível cenário.

Unicast (Figura 5) é o termo utilizado para descrever a comunicação onde uma mensagem é enviada de um ponto à outro. Neste caso, existe apenas um emissor e um destinatário por mensagem.

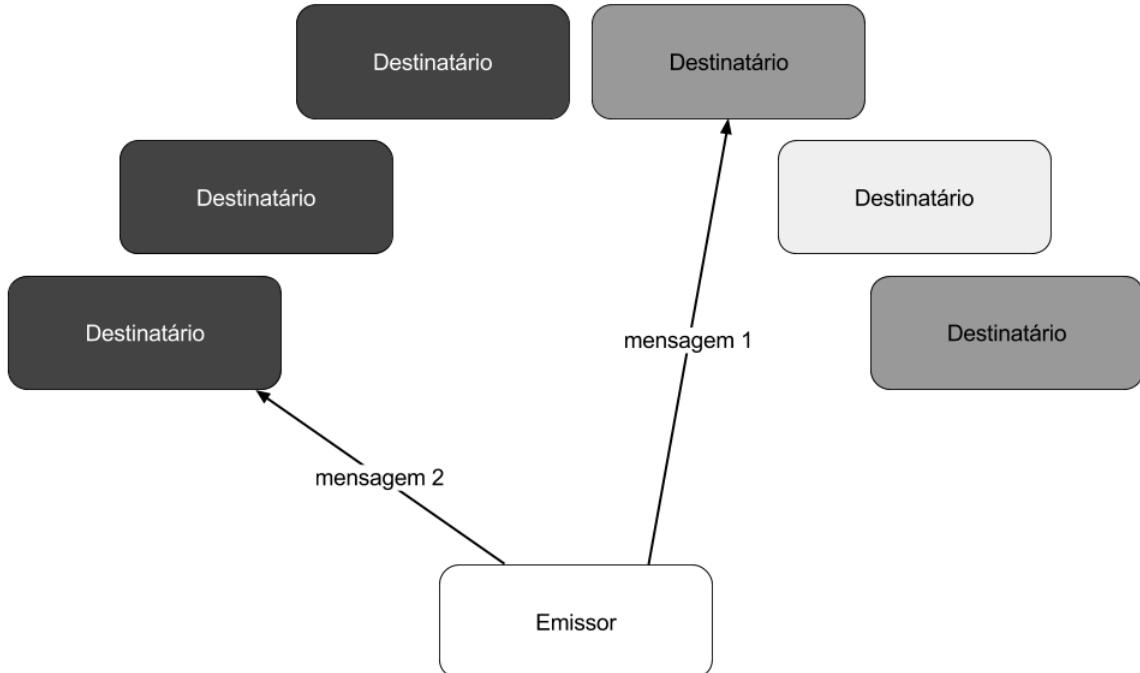


Figura 5 – *Unicast*

Fonte: Autor

Broadcast (Figura 6) representa a comunicação de uma mensagem enviada de um ponto para todos os outros pontos da rede. Neste método, existe apenas um emissor mas a mensagem é enviada para todos os destinatários conectados.

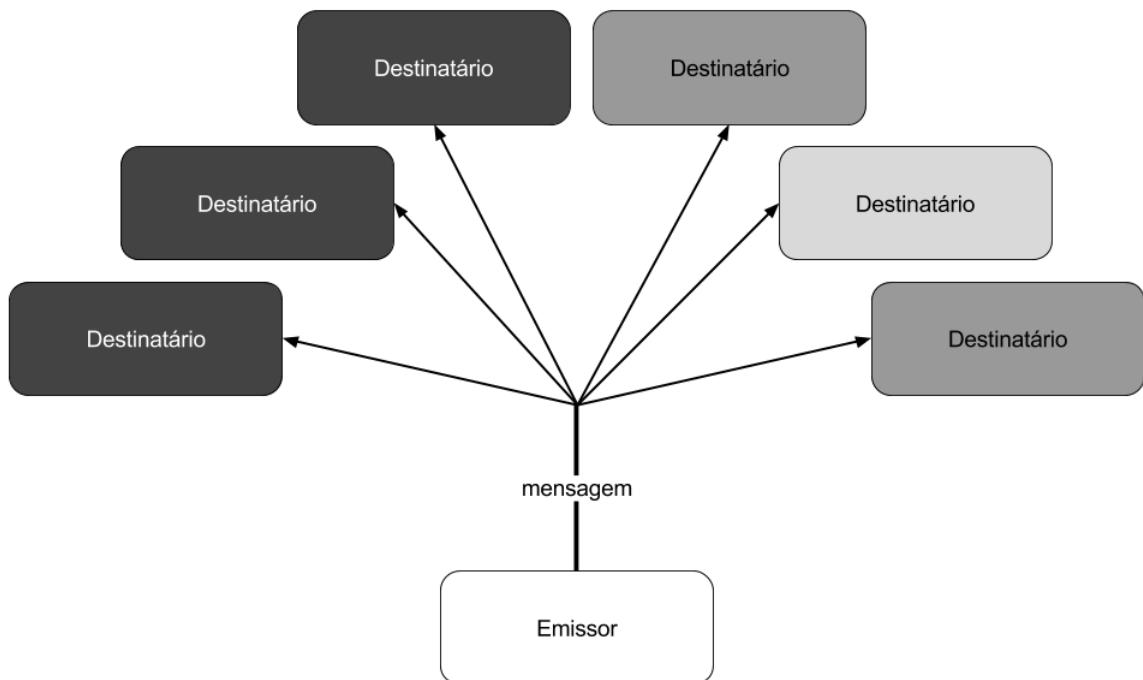
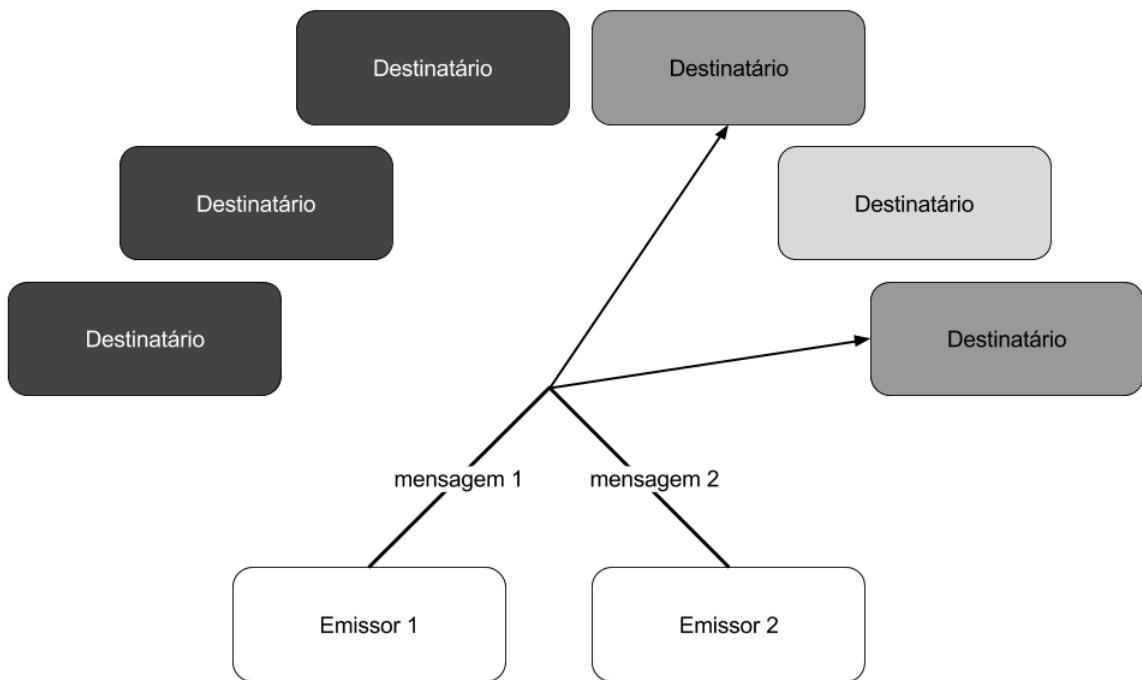


Figura 6 – Broadcast

Fonte: Autor

Multicast ([Figura 7](#)) descreve a comunicação onde uma mensagem é enviada de um, ou mais pontos, para um conjunto de outros pontos. Neste método, pode-se ter um ou mais emissores e a mensagem é distribuída para um conjunto de destinatários (pode não existir nenhum destinatário ou qualquer número de destinatários).

Figura 7 – *Multicast*

Fonte: Autor

2.4 Redes LAN

As redes de comunicação têm geralmente duas categorias primárias, que podem ser diferenciadas pelo seu tamanho e área de cobertura. Existem as redes locais (LAN ou *Local Area Networks*) e as redes de longo alcance (WAN ou *Wide Area Networks*). As redes metropolitanas (MAN ou *Metropolitan Area Networks*) se encaixam como categoria entre as LANs e WANs.

As redes locais (LANs) são o tipo mais comum de redes, podendo ser encontradas na maioria das residências, empresas e instituições de ensino. É comumente privada e limitada a alguns poucos quilômetros.

As LANs atuais são redes de alta velocidade (entre 100 Mbps a 10 Gbps), baixo índice de erros e cobrem uma área geográfica relativamente pequena (Tanenbaum, 2010). As LANs conectam estações de trabalho, periféricos, roteadores, *switches* e outros dispositivos em um único prédio ou área geográfica. O padrão *Ethernet* é de longe o tipo mais comum de LAN cabeada.

2.5 Família de padrões IEEE 802

O conjunto de padrões que lida com as redes LAN é o IEEE 802. Este padrão trata sobretudo das camadas física e de enlace do modelo OSI aplicadas em redes locais.

O IEEE através do padrão referido subdividiu a camada de enlace em duas subcamadas nomeadas de LLC (*Logical Link Control* ou Controle de Enlace Lógico) e MAC (*Media Access Control* ou Controle de Acesso ao Meio).

A subcamada de Controle de Enlace Lógico se localiza abaixo da camada de Rede do modelo OSI. Esta é responsável pelo processo de encapsulamento e desencapsulamento da camada de Enlace, lidando portanto com o controle de fluxo e erros. O cabeçalho criado pelo LLC é quem informa à camada de enlace o que fazer com o pacote quando esta recebe o *frame*. Por exemplo, quando um *host* recebe um *frame*, este olha para o cabeçalho gerado pelo LLC para entender que o pacote está destinado ao protocolo IP na camada de Rede. A subcamada também é responsável pela multiplexação¹ e decodificação de protocolos na camada de enlace.

A subcamada MAC age como interface entre a camada LLC e a camada física da rede. O padrão IEEE 802.3 especifica os endereços MAC, que identificam unicamente múltiplos dispositivos da camada de enlace, como as placas de rede. A subcamada MAC mantém uma tabela de endereços MAC (ou endereços físicos) de dispositivos. Cada dispositivo deve ser um endereço MAC único para ingressar na rede.

A subcamada MAC é quem provê serviços de comunicação *Unicast*, *Multicast* e *Broadcast*.

2.6 Família de padrões IEEE 802.11

As redes Locais utilizam-se de meio cabeados e sem fio para a transmissão de bits entre os dispositivos da rede. As redes sem fio, também conhecidas como Redes *Wifi* ou *Wireless* foram uma grande novidade tecnológica dos últimos anos. A partir de 1997, o IEEE começou a definir uma série de padrões de transmissão e codificação para redes sem fio através da família de padrões 802.11, que é aninhada à família de padrões 802. Como base, o referido padrão trata de especificações ligadas às camadas de rede e de enlace do modelo OSI voltados para Redes sem fio de Área Local (WLAN ou *Wireless Local Area Networks*).

Ao longo dos anos, uma miríade de padrões foram adicionados à família 802.11, sendo diferenciados por letras subsequentes ao número da norma. Em cada padrão adicionado à família 802.11, há uma série de especificidades que adequam-se às linhas gerais da norma. Como exemplo, são citadas algumas dessas normas na Tabela 1.

¹ Combinação de dois ou mais canais de informação por apenas um meio de transmissão

Padrão IEEE	Descrição
802.11a	Padrão de comunicação sem fio baseado na técnica de multiplexação por divisão de freqüência ortogonal (OFDM). Opera na frequência 5.8 GHz e alcança velocidades de até 54 Mbps, porém entrega em média 20 Mbps. Sofre menor interferência de sinal que o padrão 802.11b e possui melhor taxa de transferência.
802.11b	Padrão de comunicação sem fio que opera na frequência 2.4 GHz usando o mesmo método de acesso ao meio do padrão 802.11 original. A faixa de frequência é a mesma utilizada em <i>bluetooth</i> , microondas, celulares e alguns rádio amadores, e por isso sofre relativa interferência. É um dos padrões mais utilizados em roteadores sem fio devido ao baixo custo. Oferece velocidades na média de 11 Mbps.
802.11e	Agrega Qualidade de Serviço (QoS) às redes 802.11. Permite a transmissão de classes de tráfego diferentes, o que otimiza a utilização da rede.
802.11g	Utiliza a faixa de 2.4 GHz do 802.11b porém usa a técnica de transmissão OFDM assim como o padrão 802.11a. Também sofre interferências assim como o padrão 802.11b.
802.11n	Utiliza técnicas de comunicação MIMO aliadas a canais de 40 MHz para entregar taxas de transferência entre 54 a 600 Mbps. Oferece suporte às frequências dos padrões 802.11a e 802.11b.
802.11p	Emenda que faz parte da família de padrões WAVE. Implementação de redes sem fio para ambientes veiculares que também utiliza a técnica de modulação de sinal OFDM.

Tabela 1 – Padrões IEEE 802.11

2.6.1 Modos de operação

As redes sem fio suportam basicamente dois modos de operação: Infra-estruturado e ad-hoc.

O modo de operação infra-estruturado utiliza de pontos de acesso (*Access Point* ou AP) para controlar os nós cobertos por determinada área geográfica. Como exemplo, na (Figura 8) os nós de rede sem fio se comunicam com outros equipamentos ou com a rede cabeadas através de um ponto de acesso, que é representado por um roteador sem fio.

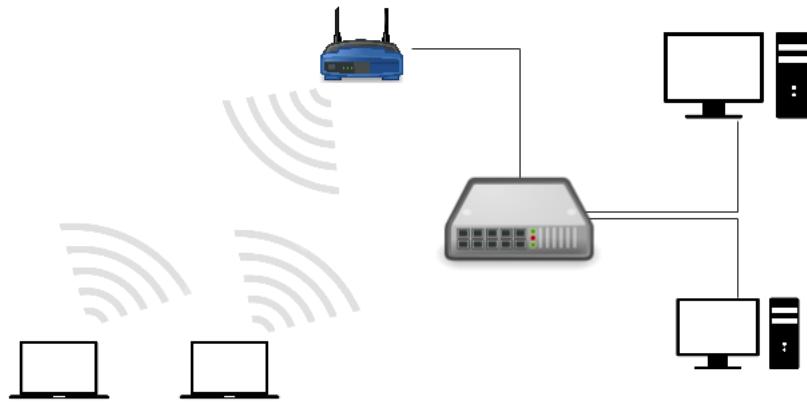


Figura 8 – Modo Infra-estruturado

Fonte: Autor

As Redes *Ad Hoc* se diferenciam por serem redes sem fio onde não há um nó central servindo como ponto de acesso. As estações ou nós comunicam-se como se fossem, ao mesmo tempo, os *hosts* e os roteadores da rede ([Figura 9](#)).

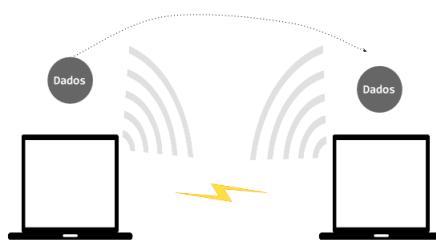


Figura 9 – Modo de Operação Ad-Hoc

Fonte: Autor

2.6.2 Arquitetura IEEE 802.11

Quando duas ou mais estações se comunicam umas com as outras, estas compartilham um Conjunto Básico de Serviços(BSS ou *Basic Service Set*). Um BSS mínimo é formado entre dois nós. As redes locais 802.11 utilizam o BSS como um bloco de construção padrão.

O BSS que não se conecta a um AP é chamado de BSS Independente (*Independent BSS* ou IBSS). Este tipo de BSS é comumente visto em Redes *Ad Hoc*. A formação de um IBSS pode ser melhor entendida através da [Figura 10](#).

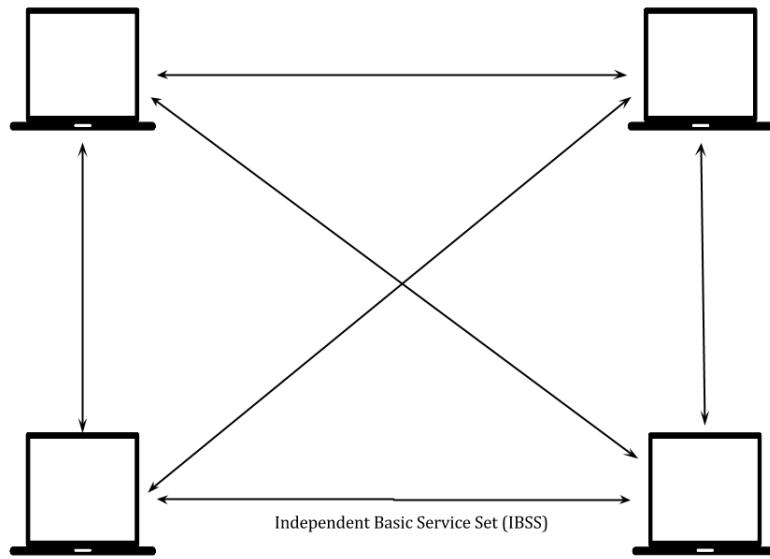


Figura 10 – Arquitetura IEEE 802.11 - IBSS

Fonte: Autor

Em redes infra-estruturadas, cada estação conectada ao AP executa o mesmo protocolo MAC e compete pelo acesso ao mesmo meio sem fio compartilhado. O BSS controla o acesso a recursos e serviços do AP, filtrando os quadros transmitidos por outras estações que não pertencem ao BSS. Um BSS pode estar isolado ou pode se conectar a outro BSS utilizando um DS (*Distribution System* ou Sistema de Distribuição). O ponto de acesso funciona como uma ponte neste caso. O conceito de DS aumenta a cobertura da rede, fazendo com que cada BSS se torne uma extensão de uma rede ainda maior.

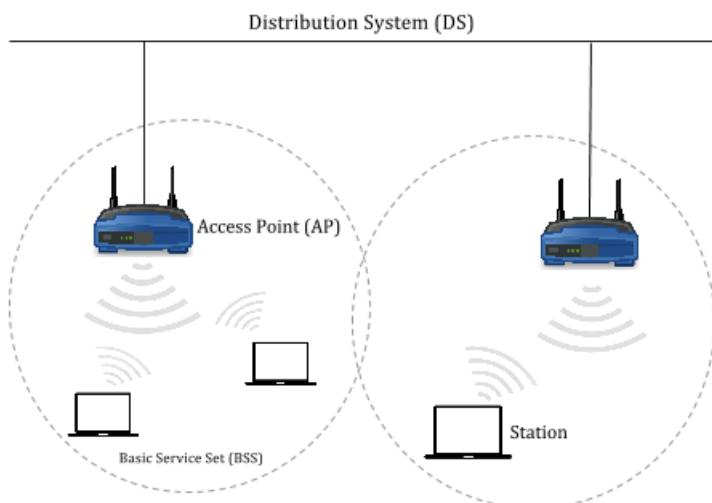


Figura 11 – Arquitetura IEEE802.11 - BSS

Fonte: Autor

Para entrar em um BSS, a estação deve primeiramente "ouvir" a frequência transmitida pelo ponto de acesso, e depois executar diversos passos que incluem processos de autenticação e associação. Os usuários de redes sem fio 802.11 identificam a presença de um BSS através de uma SSID (*Service Set Identification* ou Identificação de Conjunto de Serviços) que é anunciada pelos transmissores sem fio. As interfaces de rádio identificam um BSS no nível da subcamada MAC através do BSSID (*BSS Identification* ou Identificação BSS), que é um campo da dispositivo transmissor com formato de endereço MAC.

Cada BSS deve ser identificado por um BSSID único, que em uma rede infraestruturada é frequentemente escolhido como o endereço MAC do ponto de acesso. Em um BSS Independente, normalmente é utilizado um endereço MAC administrado localmente, formado por um número aleatório de 46 bits, e 2 bits identificando o endereço como "individual" (em vez de "grupo") e "local" (em vez de "universal"). Ademais, o padrão IEEE 802.11 define um BSSID especial, com todos os 48 bits tendo valor binário 1.

2.6.3 Serviços IEEE 802.11

O IEEE define diversos serviços que precisam ser fornecidos pela LAN para prover funcionalidade equivalente às LANs cabeadas. Os serviços mais importantes são:

O serviço de Associação estabelece uma associação inicial entre uma estação e um ponto de acesso. Antes que uma estação possa transmitir ou receber quadros em uma WLAN, sua identidade e endereço precisam ser conhecidos. Para esse fim, uma estação precisa estabelecer uma associação com um ponto de acesso. O ponto de acesso pode, então, comunicar essas informações com outros pontos de acesso para facilitar o roteamento e a entrega dos quadros endereçados.

O serviço de Reassociação torna possível uma associação estabelecida transferir-se de um ponto de acesso para outro, permitindo o deslocamento de uma estação móvel.

O serviço de Desassociação emite uma notificação por parte de uma estação ou de um ponto de acesso de que uma associação existente está terminada. Uma estação deve emitir esta notificação antes de deixar uma área ou de desligar. Entretanto, o recurso de gerenciamento do MAC se protege contra estações que desaparecem sem notificação.

O serviço de Autenticação é usado para estabelecer identidade entre estações. Em uma LAN cabeada, geralmente se considera que o acesso a uma conexão física significa autorização para se conectar com a LAN. Essa não é uma suposição válida para uma LAN sem fio, na qual a conectividade é obtida simplesmente com uma antena corretamente sintonizada. O serviço de autenticação é usado pelas estações para estabelecer sua identidade em estações com as quais ela deseja se comunicar. O padrão não exige qualquer esquema de autenticação específico, que poderia variar de um protocolo relativamente inseguro a

esquemas de criptografia de chave pública.

O **Serviço de Privacidade** é usado para impedir que o conteúdo das mensagens seja lido por outras pessoas além do destinatário pretendido. O padrão sugere o uso opcional da criptografia para garantir privacidade.

2.7 Redes Ad-hoc Móveis

As redes *ad hoc* se destacam por serem redes onde não há um nó central - como um roteador ou um *switch* - servindo como ponto de acesso. Os nós funcionam de forma similar a roteadores. Estes nós que se conectam consomem e geram dados, assim como armazenam e entregam informações. Em redes *ad hoc* são trocados quadros de arquitetura IBSS, se diferenciando das redes de infraestrutura, que utilizam quadros BSS. Além disso, as redes *ad hoc* não possuem conexão direta com a Internet. Justamente por serem redes sem infraestrutura, as mesmas não possuem provedor de acesso.

As redes sem fio de nós móveis que estão próximos uns aos outros são chamadas de MANETs (*Mobile Ad Hoc Networks* ou Redes *Ad Hoc* Móveis). Estes nós podem ser *notebooks*, *tablets*, *smartphones*, etc. Por possuírem dispositivos que se movem constantemente, as MANETs estão sujeitas à constantes variações na topologia de rede, com nós que podem entrar e sair da rede em um curto intervalo de tempo, ocasionando a alteração rápida de rotas.

O roteamento de pacotes em redes *ad hoc* é mais complexo do que em redes infraestruturadas e possui uma série de desafios. Para rotear os pacotes, os desafios encontrados são: mobilidade dos nós que impactam diretamente as rotas, implantação de Qualidade de Serviço (QoS) para os fluxos de aplicações multimídia e a escalabilidade da rede. Há também desafios com relação ao consumo de energia elétrica que os nós dispõem para realizar a transmissão e recepção dos dados além de realizar outros processos. Como a troca de informações entre os nós é intensa, a energia é gasta de maneira mais acelerada. Desafios que envolvem conectividade também preocupam. As redes *ad hoc* apresentam condições inconstantes como taxas de erro de bit variáveis e interferências na comunicação por outros meios de transmissão.

Ao longo dos anos, diversos protocolos de roteamento para MANETs foram criados. Seguindo uma primeira abordagem, os protocolos de redes *ad hoc* móveis foram classificados baseando-se na topologia (Royer & Toh, 1999) ou na posição geográfica (Giordano et al., 2004). Os protocolos baseados na topologia ainda podem ser subclassificados como pró-ativos, reativos e híbridos. Alguns protocolos são citados e categorizados na [Figura 12](#).

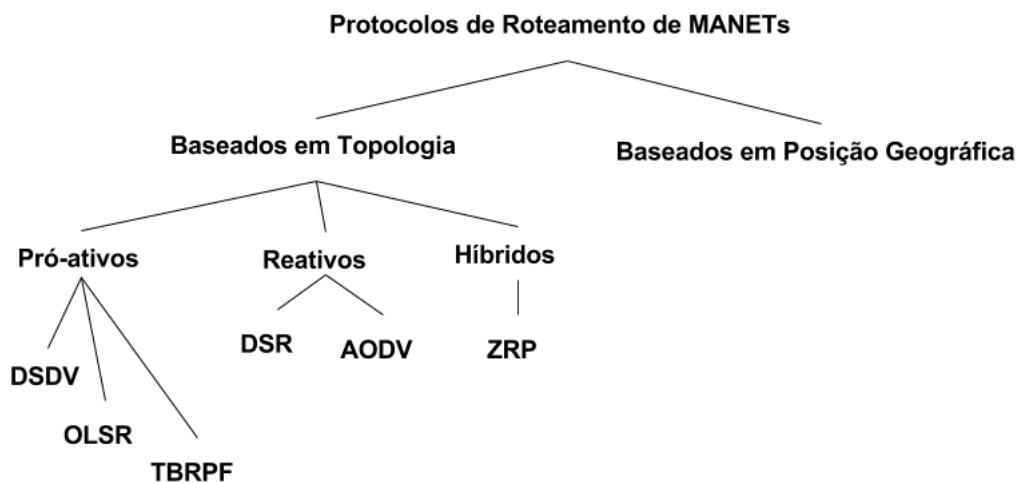


Figura 12 – Protocolos de Roteamento em MANETs

Fonte: Autor

Não é de escopo deste trabalho abordar os detalhes dos protocolos mostrados, entretanto é importante salientar que o estudo de protocolos das MANETs serviu de base para o desenvolvimento do estudo das redes veiculares, principal tema deste trabalho.

3 Redes de Comunicação Veiculares

Segundo a Organização Mundial de Saúde(OMS), acidentes de carro estão entre as 10 maiores causa de morte no mundo¹ acesso em 22 de jun. 2016. Conforme relatado pela Universidade de Stanford, mais de 90% dos acidentes têm origem em falha humana² acesso em 25 de jun. 2016. Tais dados são alarmantes para a sociedade visto que nenhum meio de transporte tira tantas vidas como o meio rodoviário.

Se comparados com transporte de outros meios, os transportes terrestres são os menos seguros. Os outros meios, como o aquático, aéreo e ferroviário, sofrem menos acidentes porém eventuais catástrofes assumem maiores proporções por justamente serem transportes de massa. Estes meios de transporte entenderam melhor a carência de segurança na locomoção e por isso há algumas décadas vemos navios e aviões equipados com sistemas de comunicação utilizando transponders, que auxiliam na prevenção de acidentes e na escolha de novas rotas, seja por tráfego denso ou por condições climáticas adversas. Nos trens, o uso de tecnologias 3G e 4G pelos sistemas de comunicação também tem sido fundamental para proporcionar o aumento da segurança e da pontualidade nas ferrovias metropolitanas e intercidades.

Atualmente, os condutores de veículos terrestres contam com itens de proteção como cintos de segurança, *airbags*, sensores de obstáculo, alarmes de velocidade, e outros equipamentos. Contudo, alguns destes itens não têm provocado um efeito de compra pelos motoristas (pelo seu preço adicional em veículos novos) e muito menos causado uma substancial diminuição no número de acidentes. Segundo prevê a OMS, acidentes automobilísticos passarão da 9^a para a 7^a maior causa de morte de humanos entre todas as idades em 2030 (*Global Status Report on Road Safety*, 2015). Por isto, governos e grandes empresas continuam a trabalhar no desenvolvimento de tecnologias para aumento da segurança veicular. O potencial advento de sistemas de comunicação entre veículos representa o próximo passo na evolução da segurança veicular, possibilitando a interação entre diferentes veículos para evitar acidentes. Segundo Yang et al(2014), os principais objetivos desses sistemas são (1) oferecer conectividade ubíqua para os usuários e (2) possibilitar a comunicação de veículo para veículo, oferecendo condições necessárias para que aplicações com diferentes requisitos sejam atendidas satisfatoriamente. Tais aplicações compõem um ITS (*Intelligent Transportation System* ou Sistema de Transportes Inteligente). Exemplos destas aplicações incluem monitoração cooperativa do tráfego, controle de fluxos de tráfego, auxílio a cruzamentos sem sinalização, prevenção de colisões, propagação de mensagens de emergência, serviços de informação próximos, etc.

¹ <http://www.who.int/mediacentre/factsheets/fs310/en/>

² <http://cyberlaw.stanford.edu/blog/2013/12/human-error-cause-vehicle-crashes>

Os sistemas de comunicação entre veículos formam as chamadas redes veiculares. Estas redes são compostas por veículos e equipamentos fixos geralmente localizados às margens de estradas ou de ruas. As redes veiculares se diferenciam de outras redes sem-fio principalmente pela natureza dos nós, que são compostos por veículos com interfaces de comunicação sem-fio, e por equipamentos fixos nos entornos das vias.

As redes veiculares possuem um conjunto de desafios para sua adoção em larga escala. De início, percebemos que os nós destas redes apresentam alto grau de mobilidade e limitam-se a trajetórias em vias públicas de acesso. Além disso, verifica-se o dinamismo dos cenários e a escalabilidade em termos do número de nós na rede. A perda de conectividade durante a transmissão de dados e o tempo reduzido em que os nós permanecem em contato são outros desafios.

Neste capítulo, veremos o desenvolvimento das Redes Veiculares e os padrões que foram criados para sua eventual consolidação de mercado.

3.1 História

Os primeiros estudos em comunicação veicular iniciaram no começo dos anos 80 quando o governo do Japão instou a JSK (Associação de Tecnologia Eletrônica de Tráfego de Automóvel e de Condução) a iniciar estudos sobre o desenvolvimento de Sistemas de Transporte Inteligentes. Alguns anos depois, o congresso americano ordenou a criação de um programa chamado IVHS (*Intelligent Vehicle Highway System* ou Sistema de Rodovia Veicular Inteligente) baseado no ato pela Eficiência dos Transportes de Superfície (*Intermodal Surface Transportation Efficiency - ISTEA*) de 1991. Depois, os programas de pesquisas americano PATH (da Universidade da Califórnia) e europeu CHAUFFEUR demonstraram técnicas para acoplamento eletrônico de veículos.

No início dos anos 2000, um projeto europeu chamado CarTALK2000 foi lançado com objetivo de investigar soluções relacionadas a condução confortável e segura baseada em comunicação interveicular. Nos últimos anos, sendo impulsionados pelos padrões de interfaces sem fio IEEE 802.11, o número de pesquisas na área de comunicação interveicular aumentou consideravelmente. Além disso, grandes empresas automobilísticas começaram a se interessar pelo assunto formando um consórcio sem fins lucrativos chamado Car2Car que tem por objetivo reforçar a eficiência e segurança no tráfego rodoviário, utilizando-se da comunicação interveicular (V2V) e contando com o amparo de infraestruturas de acostamento (V2I). Empresas como Audi, BMW, Ford, Hyundai, Jaguar, Renault, Volkswagen, Volvo e Yamaha são membros deste consórcio.

3.2 VANETs

Nos últimos anos, com o avanço no estudo das MANETs, foi iniciado o estudo em massa das redes móveis no campo interveicular. As VANETs começaram então a ser desenvolvidas, funcionando como um tipo de MANET de larga escala, tornando veículos em nós de rede sem fio. No entanto, algumas características diferenciam as VANETs das MANETs, tais quais:

- Nas VANETs([Figura 13](#)), há uma maior restrição das trajetórias em que os nós podem percorrer, visto que os veículos só podem trafegar ruas, avenidas e rodovias.
- Não há problemas com relação à energia, já que a energia não é um fator crítico em automóveis, como pode ser para outros tipos de sensores.

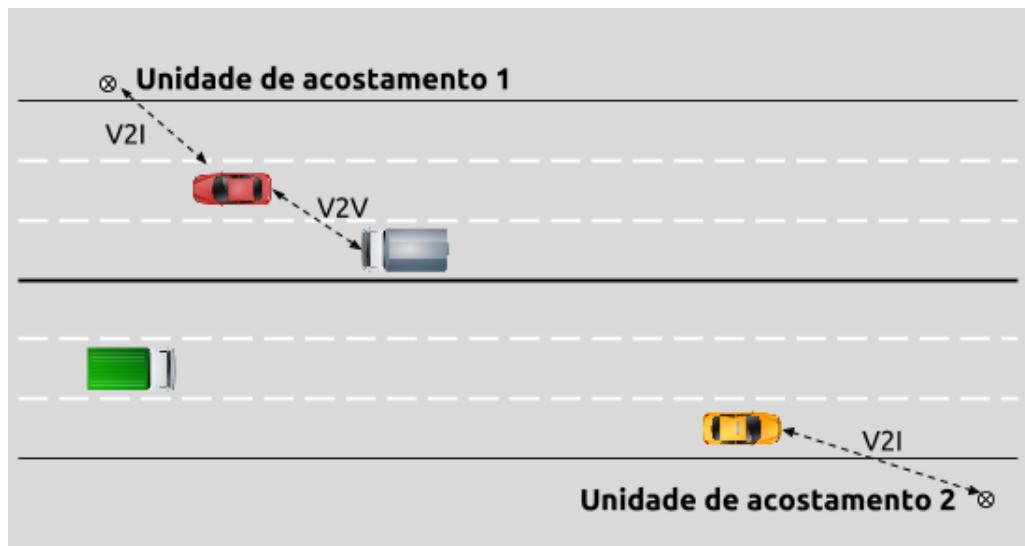


Figura 13 – VANETs

Fonte: Autor

Com a maior atenção de pesquisadores, o estudo das VANETs foi se aprimorando, de maneira que as organizações responsáveis pela padronização de tecnologias montaram equipes para discutir normas que regimentassem a forma com que as comunicações veiculares seriam utilizadas.

A normalização das redes veiculares iniciou-se em 2006, com o desenvolvimento da família de padrões WAVE. Estes padrões serão vistos em maior detalhe nos próximos capítulos.

3.3 Dispositivos Car2X: Componentes e Tipos de Comunicação

Um dispositivo Car2X é um meio de comunicação de rádio que pretende oferecer serviços interoperáveis para os transportes. Funciona como um “roteador” para veículos. Pode apenas transmitir ou receber de dados (*Simplex*) ou realizar ambas as operações.

Atualmente, várias empresas produzem hardware, *firmware* e software para dispositivos de tecnologia Car2X. Entre elas, se destacam a Cohda Wireless, Autotalks, Commsignia, Rohde & Schwarz, Vector Informatik e Unex.

Os dispositivos Car2X produzidos atualmente por estas empresas são unidades de bordo e unidades de acostamento.

As **unidades de bordo** são dispositivos de rede veicular usados por objetos móveis no trânsito. Podem ser desde veículos até unidades portáteis de sinalização ou unidades para pedestres (como trabalhadores em uma rodovia). Também são chamadas de *On Board Equipments* (OBE) ou *On-Board Units* (OBU). Podem se comunicar com outros dispositivos Car2X estáticos ou móveis. A figura [Figura 14](#) mostra um exemplo de unidade de bordo produzida pela empresa Cohda Wireless.



Figura 14 – OBU produzida pela Cohda Wireless

Fonte: <http://cohdawireless.com/Products/Hardware> acesso em 22 de set. 2016

As **unidades de acostamento** são dispositivos de rede veicular estáticos instalados ao lado de rodovias ou avenidas (num poste de luz, semáforo ou ponto de pedágio). Também chamados de *Road Side Equipments* (RSE) ou *Road Side Units* (RSU). Podem estar conectadas a uma rede de infraestrutura. Podem se comunicar com outros dispositivos Car2X estáticos ou móveis. A figura [Figura 15](#) mostra um exemplo de unidade de acostamento produzida pela empresa Cohda Wireless.



Figura 15 – RSU produzida pela Cohda Wireless

Fonte: <http://cohdawireless.com/Products/Hardware> acesso em 22 de set. 2016

Os dispositivos Car2X têm duas principais formas de comunicação: veículo para veículo e veículo para infraestrutura.

As comunicações de **veículo para veículo** (*Vehicle to Vehicle* ou V2V) são transferências de informação apenas entre unidades de bordo de veículos. O protocolo WSMP é o que provê conexões para este tipo de comunicação no padrão IEEE 1609 WAVE.

As comunicações de **veículo para infraestrutura** (*Vehicle to Infrastructure* ou V2I) são as trocas de informações entre unidades de bordo e unidades de acostamento. É possível que uma unidade de acostamento conecte-se a uma rede de infraestrutura, o que poderia prover conexão à Internet as unidades de bordo próximas.

A Figura 16 exemplifica um cenário de comunicações V2V e V2I contendo OBUs e RSUs.

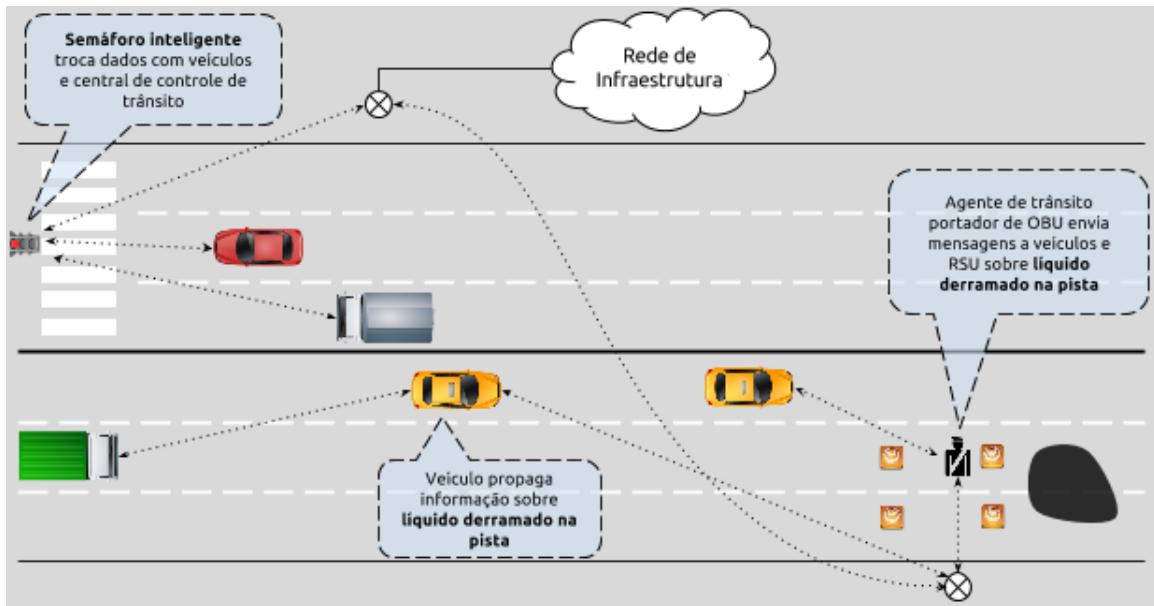


Figura 16 – Diversas OBUs e RSUs

Fonte: Autor

3.4 Família de padrões WAVE

A padronização das redes veiculares tem sido feita por dois grandes grupos: pelo IEEE, através da colaboração entre a Sociedade de Tecnologia Veicular e o Comitê de Sistemas de Transportes Inteligentes, e pelo ETSI (*European Telecommunications Standards Institute* ou Instituto Europeu de Padrões de Telecomunicações), através de seu Comitê de Transportes Inteligentes. Cabe ressaltar que este trabalho seguirá as diretrizes dos padrões IEEE 1609, visto que as normas ETSI são voltadas a atender o mercado europeu.

A família de normas IEEE destina-se a prover um conjunto padronizado de interfaces para que diferentes fabricantes de automóveis possam fornecer comunicações V2V e V2I, garantindo a interoperabilidade entre os dispositivos fabricados.

Para ajustes na camada física e de enlace lógico, o IEEE adicionou uma emenda à família de padrões 802.11 (denominada 802.11p) para especificar as mudanças nas configurações de redes sem fio adaptando-se ao meio veicular. Esta emenda visa realizar o estabelecimento da conexão mais rápida entre os nós da rede, visto que as altas velocidades de deslocamento dos nós no meio veicular demandam um tempo mais curto para a conexão.

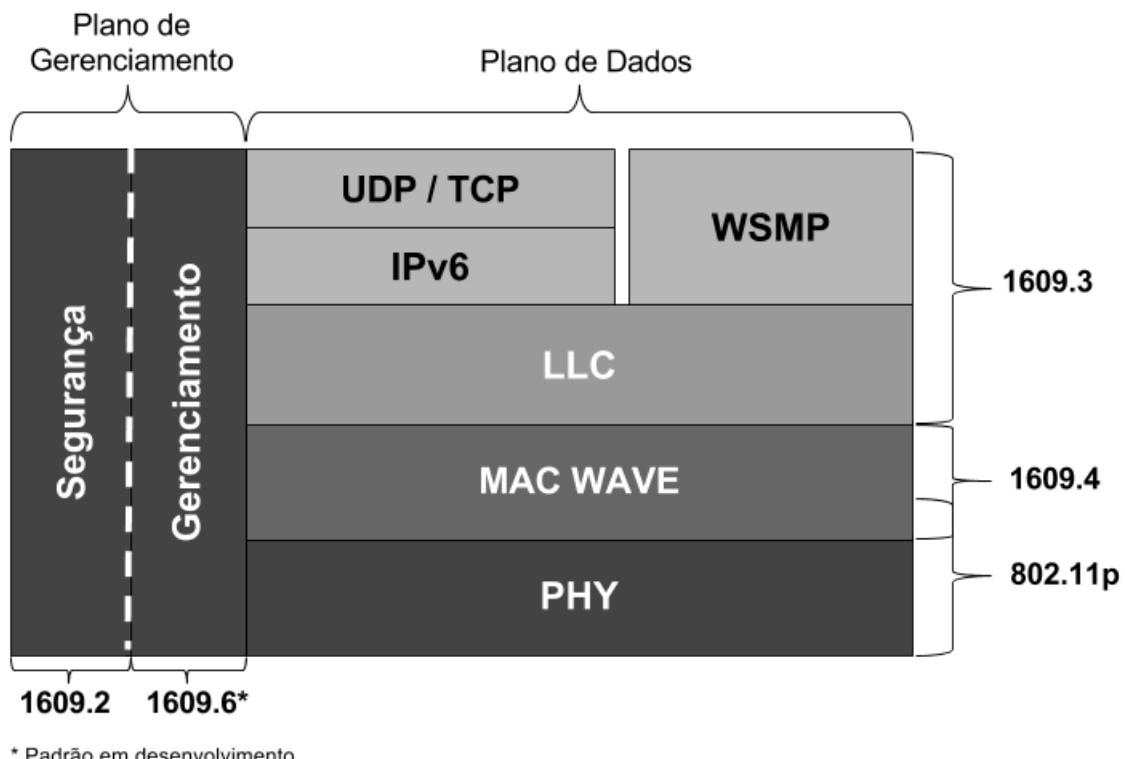


Figura 17 – Arquitetura WAVE - Planos

Fonte: Autor

A Figura 17 mostra a divisão de camadas do padrão WAVE em dois componentes abstratos, chamados de Plano de Gerenciamento e Plano de Dados. É necessário delimitar o escopo de atuação de cada um destes componentes.

O Plano de Gerenciamento é um componente abstrato da arquitetura WAVE que contém funções que gerenciam entidades no Plano de Dados. Estas funções atuam de forma indireta no transporte de dados de aplicação. O Plano de Gerenciamento pode utilizar camadas inferiores do Plano de Dados a fim de transferir informações de Gerenciamento entre os dispositivos WAVE (Maiores detalhes sobre o plano de gerenciamento na [seção 3.5](#)).

O Plano de Dados é também um componente abstrato da arquitetura WAVE que possui entidades que trocam dados. O Plano de Dados contém um conjunto definido de protocolos de comunicação que transportam dados de gerenciamento e aplicação que podem ser transferidos entre dispositivos WAVE.

Também são evidenciadas na figura a divisão de normas para as camadas da pilha WAVE. Estes padrões serão abordados ao longo deste trabalho.

Além das camadas, alguns padrões da família 1609 definem outras características, como segurança e gerenciamento para aplicações DSRC (*Dedicated Short Range Communications* ou Comunicações Dedicadas de Curto Alcance). Na [Tabela 2](#), a família de padrões

WAVE é detalhada.

Padrão	Área	Descrição	Status
802.11p	Camadas Física e MAC	Emenda que descreve as diferenças específicas na camada física e controle de acesso ao meio em ambientes de comunicação WAVE	Ativo
1609.0	Arquitetura	Descreve a arquitetura e os serviços necessários para dispositivos WAVE multicanal	Ativo
1609.1	Gerenciador de Recursos	Aplicação retirada que permitia a sites remotos se comunicarem com OBUs através de RSUs.	Retirado
1609.2	Serviços de Segurança	Define métodos para tornar as mensagens de gerenciamento e aplicação seguras. Descreve funções administrativas necessárias para suporte de funções de segurança.	Ativo
P1609.2a	Emenda à 1609.2 para gerenciamento de certificados	Descreve como os serviços de segurança WAVE respeitam a privacidade do usuário. Provê características adicionais de gerenciamento de certificação.	Em desenvolvimento
1609.3	Serviços de Rede	Especifica protocolos das camadas de rede e transporte e serviços que suportam operações multicanal.	Ativo
1609.4	Operação Multicanal	Especifica as funções e serviços da subcamada de Controle de Acesso ao Meio (MAC) que suportam operações multicanal.	Ativo
1609.5	Gerente de Comunicação	Padrão previsto para definir serviços de gerenciamento de Comunicação no suporte à conexão entre OBUs e RSUs.	Rascunho
P1609.6	Serviços de Gerenciamento Remoto	Padrão em desenvolvimento que define formatos específicos de mensagens de gerenciamento e de dados para dispositivos WAVE remotamente controlados.	Em desenvolvimento
1609.11	Protocolo de pagamentos eletrônicos	Define um nível básico de interoperabilidade técnica para equipamentos de pagamento eletrônico. Ex.: OBUs e RSUs usando DSRC	Ativo
1609.12	Alocações de Identificadores	Especifica a alocações de identificadores WAVE.	Ativo

Tabela 2 – Família de Padrões WAVE

3.5 Plano de Gerenciamento

As camadas de gerenciamento e segurança em conjunto formam o que é chamado de forma abstrata de Plano de Gerenciamento WAVE.

Os serviços do plano de gerenciamento estão associados com várias entidades do plano de dados para desempenhar funções específicas de camadas que são necessárias à operação do sistema WAVE.

3.5.1 Camada de Gerenciamento

A camada de Gerenciamento contém um grupo de entidades conceituais que coordena as operações e as chamadas de funções no Plano de Dados. Estas entidades são citadas na [Tabela 3](#).

Camada	Entidades	Descrição	Padrões
Física	PLME	Entidade que coordena conjunto de funções da camada física e contém interfaces de serviço.	802.11
Enlace - MAC	MLME	Entidade que coordena conjunto de funções da subcamada MAC e contém interfaces de serviços. Como exemplo, controla o acesso a canais de rádio específicos. Possui MIB.	802.11
	MLMEX	Entidade estendida que coordena outro conjunto de primitivas da subcamada MAC.	1609.4
Enlace - LLC	WME	Provê funções de Serviços de Rede WAVE. Fornece interfaces de gerenciamento para as entidades do plano de dados.	1609.3
Rede			
Transporte			
Aplicação	Gerenciamento de Camadas Superiores	Entidades como SDEE trabalham com primitivas na camada de aplicação.	1609.0
Segurança	Gerenciamento de Serviços de Segurança (SSME)	Entidade responsável pelo gerenciamento as informações de certificados em um dispositivo.	1609.2

Tabela 3 – Entidades de Gerenciamento

As entidades WME e MLMEX (*MLME Extension* ou Extensão MLME) herdam aspectos da SME (*Station Management Entity* ou Entidade de Gerenciamento de Estação), estabelecida no padrão 802.11.

3.5.2 Camada de Segurança

Também residindo no plano de gerenciamento, a camada de segurança pode ser chamada pela WME ou por outras entidades de camadas superiores.

O escopo da camada de segurança em dispositivos WAVE é discutido no padrão IEEE 1609.2 - Serviços de Segurança para Aplicações e Mensagens de Gerenciamento, que tem os propósitos de estabelecer a privacidade nas comunicações WAVE e proteger os dispositivos WAVE de ataques e de vazamento de informações.

Segundo a intenção dos padrões IEEE 1609, qualquer dispositivo WAVE deve proteger a privacidade de seus dados, de modo que partes não-autorizadas não conheçam dados pessoais e partes autorizadas conheçam apenas se consentidas para tal.

De modo detalhado, a norma IEEE 1609.2 descreve os seguintes serviços de segurança para aplicações e mensagens de gerenciamento:

- Serviços de Segurança Internos:
 - Serviços de Dados Seguros (SDS)
 - Gerenciamento Seguro
- Serviços de Segurança de Camadas Superiores:
 - Entidade de Verificação de Listas de Revogação de Certificado
 - Entidade de Distribuição de Certificados Ponto-a-Ponto

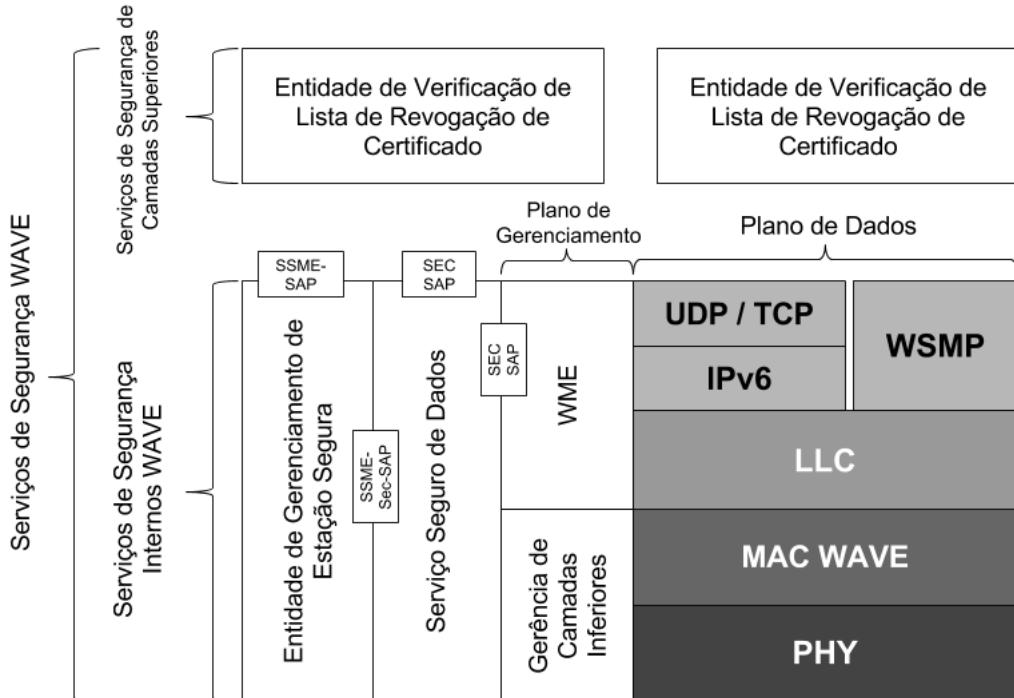


Figura 18 – Segurança WAVE Detalhada

Fonte: Autor

A camada de segurança WAVE gerencia a validação de certificados de segurança, a criação de unidades de protocolo de dados seguros (SPDU ou *Secure Protocol Data Unit*), além de operações de criptografia. Nas operações de criptografia, a norma cita algoritmos de assinatura, algoritmos hash, algoritmos de chave simétrica e assimétrica, além de diferentes abordagens de chaves de criptografia.

Não é de objetivo deste trabalho monográfico discorrer de maneira detalhada sobre o estudo de segurança das redes WAVE, e sim citar a existência da camada e realizar uma abordagem geral.

3.5.3 Interfaces

Para comunicação entre diversos protocolos no Plano de Dados, é necessário ter interfaces. As interfaces nos componentes WAVE são chamados de Pontos de Acesso de Serviços (*Service Access Points* ou SAP). Assim como no modelo OSI, os SAPs do plano de dados se localizam conceitualmente entre uma camada OSI e outra adjacente. No plano de gerenciamento, os SAPs podem ser acessados por quaisquer entidades.

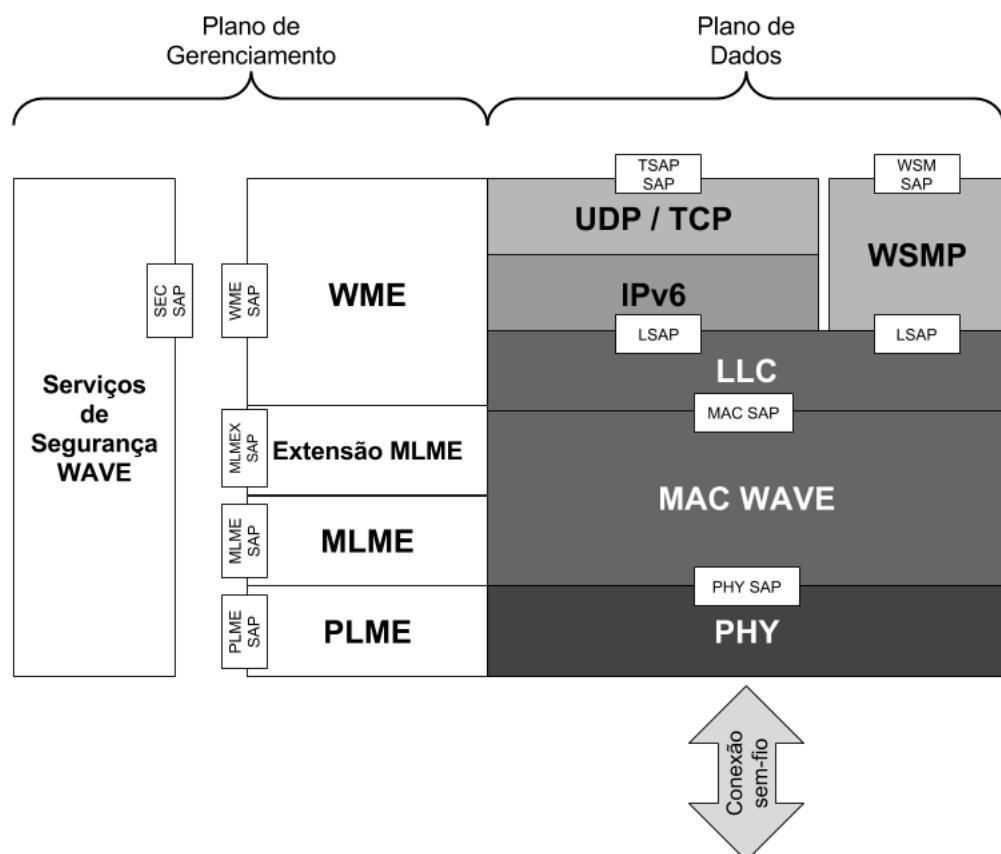


Figura 19 – *Service Access Points*

Fonte: Autor

Os SAPs são compostos de primitivas de serviço. Estas primitivas informam ao serviço como desempenhar uma determinada ação. Primitivas de serviço se parecem com procedimentos em linguagem de programação. As primitivas de serviço são “codificadas” na notação ASN.1 (Ver [Apêndice A](#)).

3.6 Camada Física

A camada física do modelo WAVE foi descrita pelo padrão 802.11p, que é uma emenda à família de padrões 802.11. Esta emenda propõe uma adaptação das redes sem fio em suas camadas física e de enlace (subcamada MAC) para o ambiente veicular.

Os meios sem fio de transmissão de dados mais utilizados na norma 802.11 baseiam-se em Rádio Frequência (RF). Existem outros meios de propagação sem fio como o Infravermelho, porém são menos utilizados. As frequências de rádio empregadas para redes de comunicação sem fio estão na faixa de frequência de micro-ondas no espectro eletromagnético([Figura 20](#)), variando entre $0,3 \times 10^9$ até 3×10^{11} Hertz.

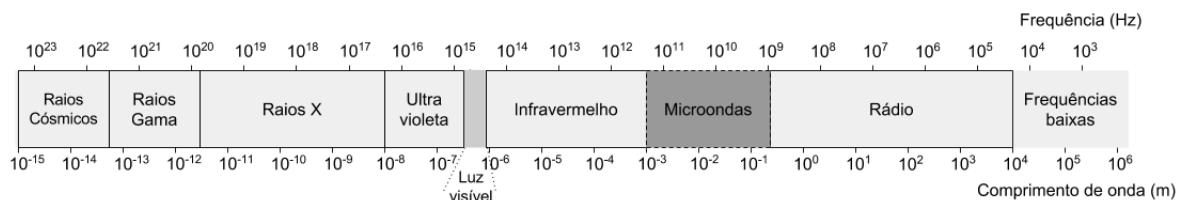


Figura 20 – Espectro Eletromagnético

Fonte: Autor

A camada física do padrão WAVE é oriunda de uma transformação da mesma camada no padrão 802.11a. Assim como o 802.11a, o 802.11p utiliza a Multiplexação por Divisão Ortogonal de Frequência (OFDM ou *Orthogonal frequency-division multiplexing*), que comprime os bits contendo os dados em ondas de rádio utilizando-se de técnicas de processamento de sinal. A técnica OFDM divide um largo espectro disponível em vários subcanais de bandas mais estreitas. Cada subcanal é utilizado para carregar parte da informação. As frequências e o tempo de transmissão de cada subcanal são diferentes a fim de que uma transmissão não interfira na outra.

O padrão 802.11p se difere do 802.11a com relação ao ambiente em que vai ser utilizado. O padrão 802.11a é voltado uso em ambientes fechados. pouca mobilidade dos nós e tem alcances curtos. Já a norma 802.11p é voltada para médias distâncias (até 1 quilômetro), alta mobilidade dos nós e mudanças rápidas de canal.

Nos Estados Unidos, as redes veiculares foram licenciadas pelo Departamento de Transporte para a operação numa largura de banda de 75MHz, operando na faixa entre

5.850 e 5.925 GHz. Este espectro havia sido separado alguns anos antes para comunicações de alcance curto(DSRC). Por isso, os termos DSRC, WAVE e 802.11p são correlacionados em diversas bibliografias e podem ser usados como sinônimos.

Na camada física, a principal emenda acrescida pela norma 802.11p foi operação das interfaces em canais de 10 MHz ao invés de 20 MHz, como eram usados em dispositivos 802.11a. A diminuição da largura do canal foi motivada por estudos feitos por Cheng et al (2008) que demonstraram que os intervalos de guarda em transmissão de 20MHz ($0.8\mu s$) não são suficientemente longos para conter os atrasos gerados por efeitos da propagação multi-percurso, que eram de até $1.5\mu s$. Um canal de 10MHz tem intervalos de guarda de $1.6\mu s$.

O espectro que segue na Figura 21 foi alocado pela Comissão Federal das Comunicações(FCC) dos Estados Unidos para o DSRC. É esperado que os sistemas WAVE desenvolvidos nos EUA trabalhem nos canais definidos pela FCC.

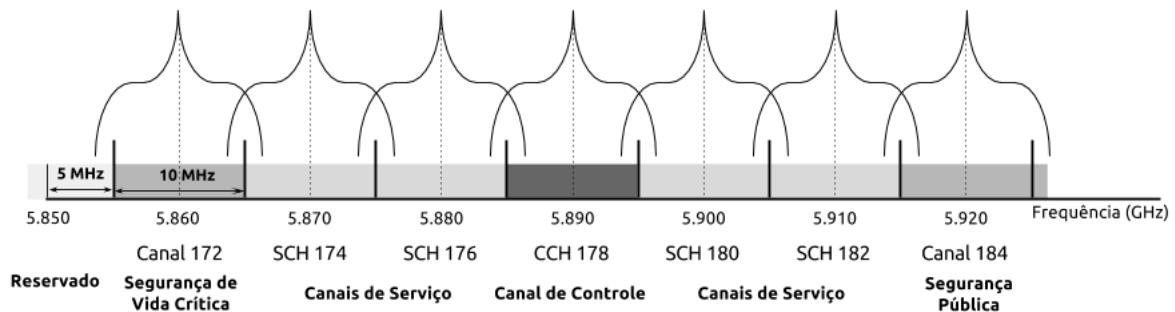


Figura 21 – Espectro WAVE FCC EUA

Fonte: Autor

Segundo o padrão 1609.0, o espetro DSRC apresenta as seguintes características no canais:

- A faixa de frequência 5850 - 5855 GHz foi colocada em reserva;
- O canal 178 é um CCH (*Control Channel* ou Canal de Controle);
- Os canais 172, 174, 176, 180, 182 e 184 são SCHs (*Service Channels* ou Canais de Serviço);
- Os canais 174 e 176 e os canais 180 e 182 podem ser combinados para produzir 2 canais de 20 MHz, nomeados de canais 175 e 181 respectivamente;
- Os canais 172 e 184 são designados para aplicações de segurança pública envolvendo segurança da vida e propriedade. Especificamente definidos através do FCC

dos EUA, o canal 172 se destina à aplicações de segurança V2V para prevenção e mitigação de acidentes e aplicações para a segurança de vida e propriedades. O canal 184, por sua vez, se destina às comunicações de longa distância e alto consumo para serem usadas por aplicações de segurança pública envolvendo segurança de vida e de propriedade, incluindo mitigação de colisões em cruzamentos.

É importante salientar que o espectro alocado pelo FCC para os EUA pode ou não ser imitado por outros países, como o Brasil. É de escopo da Agência Nacional de Telecomunicações (Anatel) determinar o espectro e a alocação dos canais para os serviços de redes veiculares com fins públicos e privados.

3.6.1 Tipos de Canais

Os padrões WAVE especificam duas tipos de canais de rádio. Os dispositivos Car2X que utilizam o padrão WAVE e que têm múltiplas camadas físicas (vários transmissores de rádio) poderão operar um canal de controle (CCH) e pelo menos um canal de serviço(SCH) ao mesmo tempo. Os dispositivos WAVE de apenas um rádio transmissor trabalham alternando entre o canal de controle e os canais de serviço. Para lidar com a limitação de um rádio transmissor, é exigido um sincronismo para que os dispositivos WAVE monitorem cada canal no seu intervalo de tempo específico.

O canal de controle é reservado apenas para mensagens do protocolo WSMP e para troca de mensagens de gerenciamento do sistema, como as mensagens de anúncio de serviços WAVE(WSA). As mensagens do protocolo WSMP podem ser recebidas por qualquer dispositivo WAVE com o canal de controle "acordado". Como exemplo, um dispositivo WAVE pode solicitar informações de sincronismo entre rádios através de uma requisição broadcast através do protocolo WSMP a um outro dispositivo WAVE próximo. O dispositivo WAVE receptor responderia com informações de sincronismo através de um quadro de gerenciamento.

Os canais de serviço são destinados a operar as transferências de dados de aplicações diversas, e também podem ser coordenados via mensagens de anúncio de serviço WAVE. Os protocolos WSMP e IPv6 podem ser usados para transmitir informações nos canais de serviço.

Para um melhor entendimento, iremos separar os diferentes tipos serviços WAVE em dois grupos: serviços “anunciados” e serviços “não anunciados”.

Os serviços WAVE “não anunciados” estão geralmente ligados à segurança pública e têm canais fixos. Os serviços “não anunciados” forçam os dispositivos WAVE recebedores a mudar para o canal proposto imediatamente após o envio de uma mensagem de gerenciamento WAVE.

Como exemplo, um veículo de segurança pública pode enviar uma mensagem do protocolo WSMP contendo informações de emergência no canal de serviço 172 (canal de segurança pública nos EUA) para que os veículos em seu percurso liberem passagem. Todos os veículos no alcance da unidade de bordo emissora processariam a mensagem imediatamente, propagando a mesma em seguida através do canal de serviço 172.

Nos serviços "anunciados", um dispositivo WAVE assume o papel de provedor. Este provedor transmite mensagens de anúncio WAVE através do canal de controle identificando e descrevendo o serviço e o canal a serem utilizados por este. Os dispositivos WAVE que decidirem se associar ao provedor sintonizam suas antenas no canal proposto pela WSA, se tornando usuários deste provedor. Assim feito, o dispositivo provedor e os usuários poderão trocar informações associadas com o serviço oferecido.

Como exemplo, a [Figura 22](#) representa um cenário no qual há 3 unidades de acostamento, todas instaladas em postes de iluminação ao longo da rodovia. Neste cenário, cada unidade de acostamento está transmitindo anúncios de serviço WAVE através do canal de controle contendo a Identificação do Serviço com o valor 0x04, atribuído ao serviço de aconselhamento de tráfego para viajantes. As unidades de acostamento enviam as mensagens de anúncio (WSAs) na forma de broadcast contendo avisos sobre o tráfego local em um canal proposto e respondem a requisições para informações adicionais de tráfego usando IPv6. Quando os dispositivos usuários entrarem na área de cobertura das unidades de acostamento, estes irão receber as WSAs e avaliarão se o valor da identificação de serviço é desejado. Os dispositivos usuários que têm interesse em aconselhamento de tráfego (neste cenário representados pelo caminhão verde e pelo carro vermelho) aceitarão este serviço e sintonizarão no canal especificado. Os dispositivos usuários que não têm interesse neste serviço irão simplesmente descartar os pacotes recebidos.

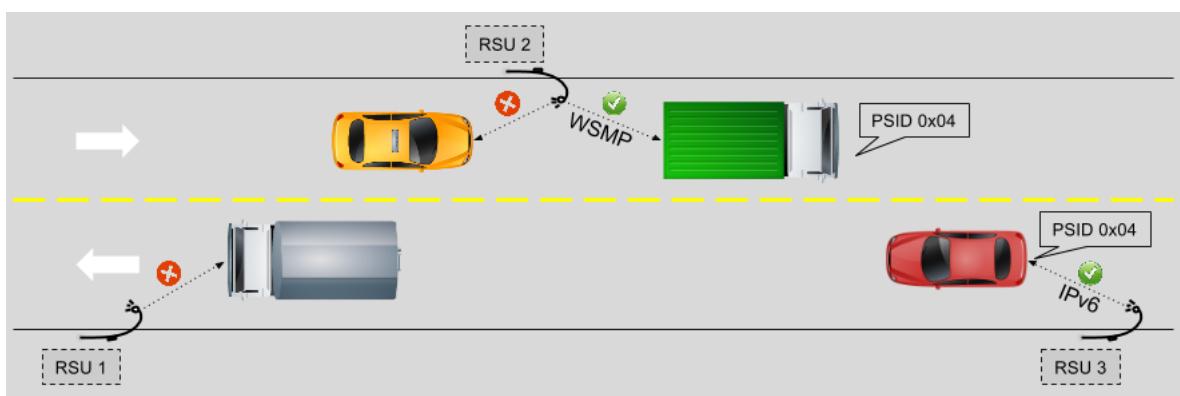


Figura 22 – Tipos de Canais

Fonte: Autor

3.7 Subcamada MAC

Na camada de enlace também foram acrescidas emendas, sobretudo na subcamada MAC. Diferentemente de uma conexão sem fio tradicional, na qual o MAC do *Access Point* está presente no cabeçalho do *frame*, a norma 802.11p define um método de troca de informações mais rápido, sem o estabelecimento de um BSSID real. Um BSSID “coringa” (com os valores binários em 1’s) é utilizado no cabeçalho dos *frames* trocados. Este é inserido no campo Endereço 3 do cabeçalho MAC (Figura 23).

Cabeçalho 802.11 MAC Comum

2 bytes	2 bytes	6 bytes	6 bytes	6 bytes	2 bytes	6 bytes	0 - 2312 bytes	4 bytes
Controle de Frame	Duração/ID	Endereço 1	Endereço 2	Endereço 3	Controle de Sequência	Endereço 4 (opcional)	Corpo do Frame	Checagem de Redundância
-	-	-	-	BSSID Exemplo: 0xB8AC6FC4 F4DC	-	-	-	-

Cabeçalho 802.11p WAVE

2 bytes	2 bytes	6 bytes	6 bytes	6 bytes	2 bytes	6 bytes	0 - 2312 bytes	4 bytes
Controle de Frame	Duração/ID	Endereço 1	Endereço 2	Endereço 3	Controle de Sequência	Endereço 4 (opcional)	Corpo do Frame	Checagem de Redundância
-	-	-	-	BSSID Coringa: 0xFFFFFFFFFFF	-	-	-	-

Campos que contém subcampos

Figura 23 – Exemplo de MAC Comum e MAC WAVE

Fonte: Autor

Além disso, a emenda adiciona à MIB (Ver mais em Apêndice B) do protocolo de gerenciamento SNMP a variável **dot11OCBActivated** como booleana. OCB (*On the Context of BSS*) significa no contexto do BSS. Caso for configurada como verdadeira e o BSSID Coringa for utilizado, a conexão entre os pares é realizada dispensando os procedimentos de associação, autenticação e confidencialidade de dados. Estes serviços então seriam de responsabilidade das camadas superiores.

Também foi adicionado através da emenda 802.11p um quadro de gerenciamento para anúncio de hora a fim de que as estações 802.11p sincronizem entre si uma referência de hora comum. A única referência definida na emenda é o padrão UTC (*Universal Time Coordinated* ou Tempo Universal Coordenado).

3.7.1 Operações multi-canal

Os dispositivos WAVE implementam operações e serviços que suportam conectividade multi-canal na subcamada de acesso ao meio (MAC), baseando-se na norma IEEE 1609.4. Esta norma descreve mecanismos efetivos que controlem a transferências de dados de camadas superiores através de múltiplos canais. Outra norma descrita nesta seção é a IEEE 802.11 MAC, que trata de forma mais abrangente a coordenação de múltiplos canais em redes sem fio.

Conforme visto na [Figura 18](#), o padrão WAVE tem uma pilha contendo protocolos, entidades de gerenciamento e serviços de segurança. As entidades de gerenciamento são importantes para o funcionamento dos protocolos pois proveem as interfaces de serviços através das quais as funções de gerenciamento de cada camada podem ser chamadas.

O padrão IEEE 1609.4 adiciona extensões ao modelo de subcamada MAC do padrão 802.11 para adaptação ao meio veicular. Para realizar operações sobre múltiplos canais sem fio tendo o parâmetro *OCBActivated* como verdadeiro, há necessidade de realizar a coordenação de canal, o que é proposto no padrão mencionado.

Na subcamada MAC, a entidade MLME (mostrada na [Figura 19](#)) é responsável por gerenciar os serviços de coordenação do canal e por auxiliar a entrega das unidades de dados de serviço MAC.

3.7.2 Serviços do Plano de Dados

Os serviços executados pela MLME são inerentes ao Plano de Dados e ao Plano de Gerenciamento. Os serviços MLME no Plano de Dados abrangem:

Coordenação de canal

A coordenação de canal pela subcamada MAC garante que os pacotes serão transmitidos no canal sem fio pretendido e permite a troca de dados entre dispositivos com operações alternadas e concorrentes em múltiplos canais. Esta última característica permite que dispositivos WAVE com uma única camada física utilizem o acesso alternado na coordenação de canal. Como exemplo, num dispositivo WAVE de única camada física os tráfegos de alta prioridade e de gerenciamento operariam no canal de controle em intervalo de tempo 0 enquanto o tráfego geral de camadas superiores atuaria no canal de serviço durante intervalo de tempo 1.

Os dispositivos WAVE de várias camadas físicas podem utilizar o acesso contínuo (a) na utilização de canal de controle ou canal de serviço, sem necessitar de alguma coordenação. Também é possível que haja acesso alternado entre os canais de serviço. A coordenação de canais é mostrada na [Figura 24](#).

Exemplos:

- (A) Um dispositivo WAVE A pode acessar o canal de controle em intervalo de tempo 0 e um canal de serviço em intervalo de tempo 1;
- (B) Um dispositivo WAVE B pode acessar um canal de serviço em intervalo de tempo 0 e o canal de controle em intervalo de tempo 1;
- (C) Este mesmo dispositivo WAVE B em algum tempo depois pode acessar um canal de serviço em intervalo de tempo 0 e outro canal de serviço em intervalo de tempo 1.

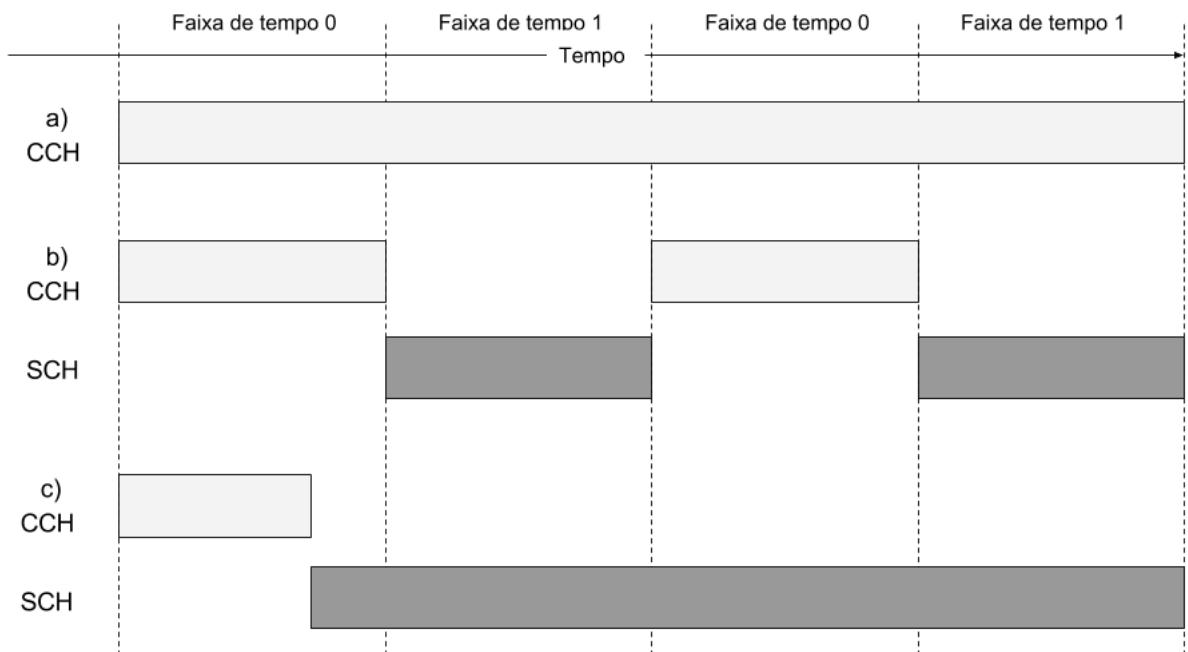


Figura 24 – Exemplos de Acesso ao Canal

Fonte: Autor

Esta alternação entre o CCH e SCH depende de um mecanismo de sincronização entre os dispositivos participantes, que é garantido pelo Serviço de Gerenciamento Sincronização multi-canal. Também é definido um período de guarda no início de cada intervalo de tempo com o objetivo de compensar os efeitos de comutação de rádio e imprecisões no sincronismo, como mostrado na [Figura 25](#).

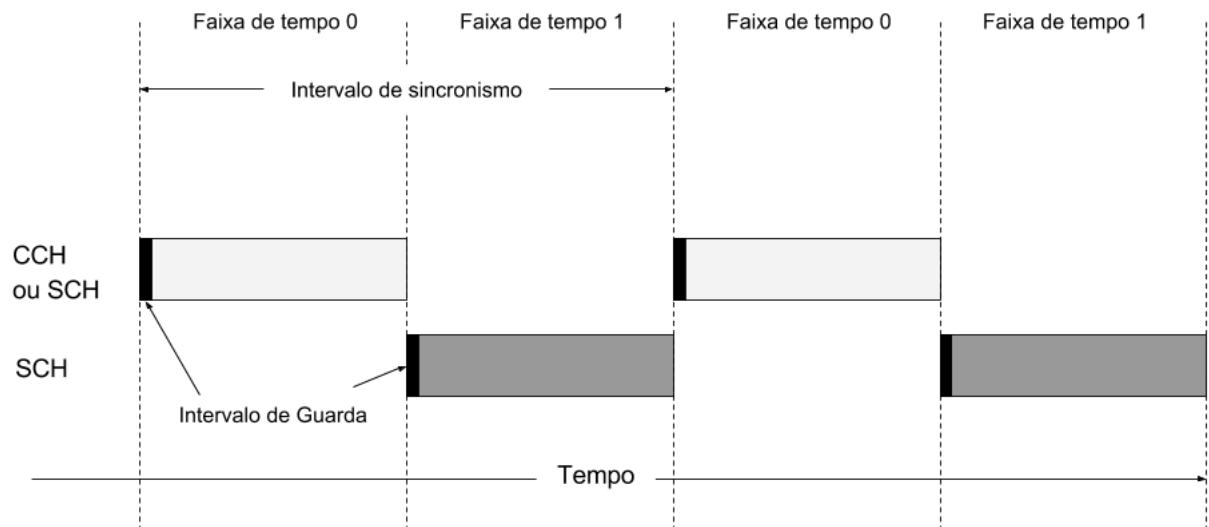


Figura 25 – Intervalo de Sincronismo e de Guarda

Fonte: Autor

Roteamento de canal

A subcamada MAC lida com dados de entrada e saída das camadas superiores, roteando os pacotes de dados de uma camada superior até o canal designado e configurando os parâmetros para transmissões sem fio WAVE. Do lado do destinatário, a subcamada MAC garante o roteamento de pacotes para o protocolo de camada superior correto.

O protocolo WSMP define o canal, a potência de transmissão e a taxa de transferência a cada mensagem. Quando dados WSMP são recebidos na subcamada MAC, os pacotes são direcionados à filas para a transmissão de acordo com o **identificador do canal** e a **prioridade**. Logo após, a subcamada MAC deve transmitir o pacote dentro do intervalo de tempo indicado, com taxa de transferência e potência de transmissão determinadas.

Datagramas IP, entretanto, têm os parâmetros canal, potência de transmissão e taxa de transferência armazenados no perfil do transmissor. Os pacotes IP são recebidos pela subcamada MAC e colocados em filas de acordo com o nível de prioridade para a transmissão no SCH. Utilizando os parâmetros estabelecidos no perfil do transmissor, os pacotes são enviados através da camada física caso a MLME delibere acesso.

Prioridades de pacotes

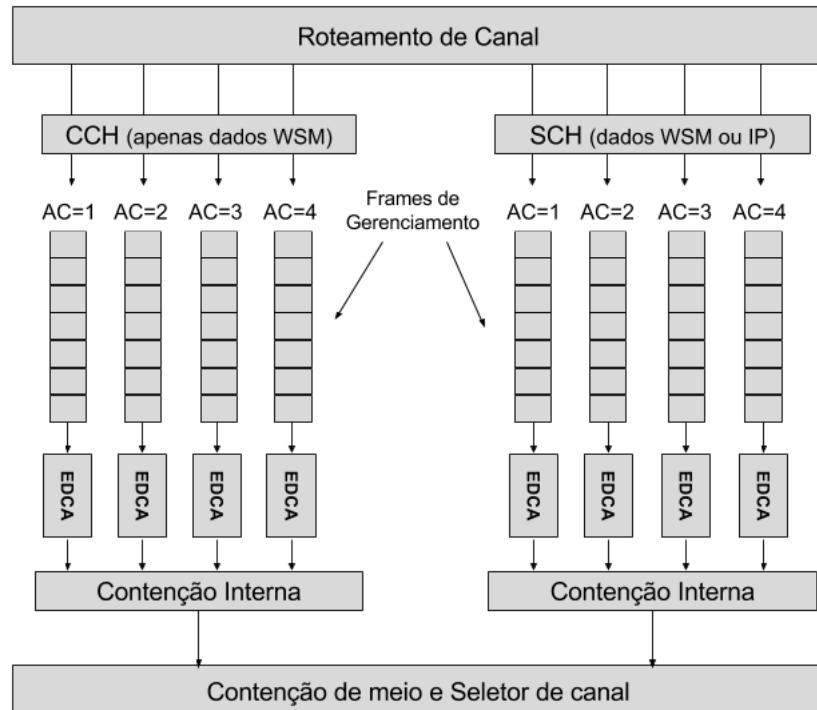


Figura 26 – MAC(Multi-Canal)

Fonte: Autor

Os padrões WAVE acomodam Qualidade de Serviço (*Quality of Service* ou QoS) nos pacotes aplicando as técnicas EDCA (*Enhanced Distributed Channel Access* ou Acesso ao Canal Distribuído Aprimorado).

Quando um pacote WSMP ou IPv6 chega na subcamada MAC, tendo o roteamento do canal apropriadamente finalizado, esta deverá “enfileirar” os dados de acordo com Categoria de Acesso (*Access Category* ou AC) indicada, que vai de 1 a 4. O mapeamento dos pacotes atribuindo o indicador de prioridade é tratado pelo padrão IEEE 802.11. Provedores de Serviço WAVE podem opcionalmente anunciar parâmetros EDCA através de um anúncio de serviço WAVE. Caso nenhum parâmetro EDCA seja especificado, o EDCA padrão é ativado.

3.7.3 Serviços do Plano de Gerenciamento

Em adição aos serviços de gerenciamento MLME mencionados no padrão 802.11 para redes sem fio em geral, o padrão WAVE IEEE 1609.4 adiciona características de acesso ao canal e sincronização de dispositivos WAVE em apoio à coordenação de canal, manutenção da MIB e reendereçamento.

Sincronização multi-canal

A entidade da subcamada MAC(MLME) utiliza uma base de tempo comum que pode ser obtida localmente ou recebida via rádio para gerar uma função de sincronismo com objetivo de alinhar os intervalos de tempo entre os dispositivos WAVE comunicantes. A MLME gera quadros de anúncio de sincronismo para distribuição de informação de sincronismo e monitora os estes quadros recebidos.

A referência de tempo comum usada para o sincronismo multi-canal é o módulo 1 segundo da UTC, utilizado por muitos equipamentos GPS.

Parâmetros como intervalo de guarda, tolerância de sincronização e tempo máximo de troca de canal são gerenciados pelo Serviço de Gerenciamento de Sincronização Multi-canal.

Acesso ao canal

A MLME controla o acesso aos canais de rádio específicos em suporte às requisições recebidas da entidade de gerenciamento WAVE (WME ou *WAVE Manager Entity*). Opções de acesso a canal incluem acesso contínuo, acesso alternado e acesso imediato, evidenciado pela [Figura 24](#).

(A) Acesso contínuo: o canal designado fica disponível por duração indefinida;

- Um dispositivo WAVE configurado para operar em um único canal de serviço ou de controle.

(B) Acesso alternado: o acesso é alternado entre os canais de serviço ou entre o canal de controle e um canal de serviço;

- Um dispositivo WAVE configurado para monitorar anúncios de serviço no canal de controle. Quando o canal de controle recebe um anúncio de serviço WAVE de seu interesse, o dispositivo o alterna para o canal de serviço durante do intervalo de tempo 1 para participar da aplicação de interesse, retornando ao canal de controle durante intervalo de tempo 0 para monitoramento de outras WSAs ou qualquer outra atividade. Alternativamente, um dispositivo pode ser configurado para alternar entre canais de serviço também.

(C) Acesso imeditado: o acesso é mudado por tempo determinado ou indefinido;

- Um dispositivo é configurado para monitorar um canal de controle. Ao receber uma WSA com uma aplicação de interesse, o dispositivo alterna imediatamente

para um canal de serviço por um tempo específico a fim de completar determinada atividade. Ao término desta atividade, o dispositivo retorna ao primeiro canal.

Outros serviços IEEE 802.11

A MLME permite outros serviços 802.11 que podem ser chamados por cada canal independentemente.

Manutenção da Base de Gerenciamento de Informação (MIB)

A MLME mantém uma Base de Gerenciamento de Informação contendo informações de configuração e status.

Reendereçamento

A MLME permite alterações dinâmicas no endereço MAC a fim de suportar pseudonimato. O pseudonimato é uma propriedade em que identidades e padrões de comportamento de entidades não podem ser deduzidos pelo tráfego de rede, sendo apenas observáveis por partes propriamente autorizadas.

3.8 Subcamada LLC

Definida no padrão 1609.3, a subcamada de Controle de Enlace Lógico do modelo WAVE está imediatamente acima da subcamada MAC. A subcamada LLC do modelo WAVE é parte essencial no direcionamento dos dados para processamento nas camadas superiores.

Nesta camada, é utilizado o protocolo SNAP. No cabeçalho deste protocolo há o campo *Type* que revela o valor *Ethertype*. Este valor indica qual protocolo da camada imediatamente acima será introduzido. A [Figura 27](#) mostra o cabeçalho do protocolo SNAP juntamente ao cabeçalho LLC em mais detalhes.

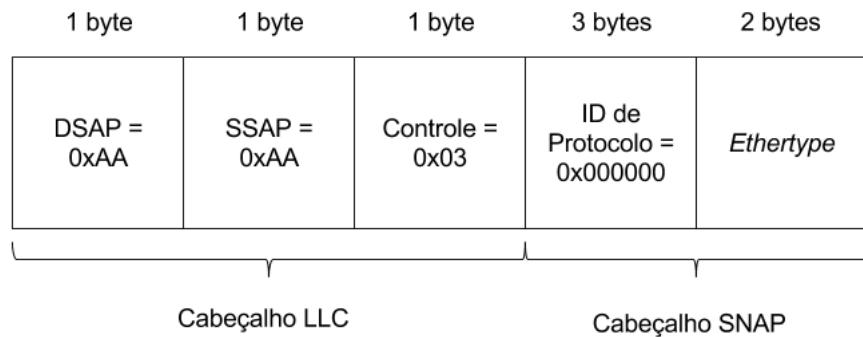


Figura 27 – Cabeçalho LLC

Fonte: Autor

Iniciando pelo cabeçalho do LLC, os campos de 1 byte DSAP e SSAP são sempre configurados com o dígitos hexadecimais 0xAA (valor binário 10101010) indicando encapsulamento.

O campo de 1 byte **control** é utilizado para verificar se os dados são sequenciais ou não. São sempre configurados com os dígitos hexadecimais 0x03 (valor binário 00000011) indicando o uso de transporte de dados sem conexão e que os dados não precisam ser sequenciados ou reconhecidos.

No cabeçalho do protocolo SNAP, o campo **Protocol ID** ou também chamado de OUI (*Organization Unique Identifier*) tem tamanho de 3 bytes e é configurado com os dígitos hexadecimais 0x000000 indicando que o protocolo a ser utilizado na camada acima é do tipo *Ethernet* (**Ethertype**).

O campo **Type** é o campo que indica qual protocolo será utilizado na camada imediatamente acima, isto é, a camada de rede. O tamanho do campo é de 2 bytes. No caso dos padrões WAVE, apenas dois protocolos são possíveis na camada de rede: IPv6 ou WSMP.

O processamento da transmissão de um pacote da camada de rede funciona da seguinte maneira:

O protocolo da camada de rede requisita a transmissão de um pacote e a subcamada LLC configura o campo **Type** ao valor *Ethertype* correspondente.

- Se o protocolo for WSMP, uma primitiva chamada DL-UNITDATAx.request é acionada e o valor *Ethertype* atribuído no campo Type é 0x88DC. Os dados são transmitidos para as camadas inferiores usando a primitiva MA-UNITDATAx.request. Se o protocolo for IPv6, uma primitiva chamada DL-UNITDATA.request é acionada e o valor Ethertype atribuído no campo Type é 0x86DD. Os dados são transmitidos

para as camadas inferiores usando a primitiva MA-UNITDATA.request.

- Se o protocolo for IPv6, uma primitiva chamada DL-UNITDATA.request é acionada e o valor *EtherType* atribuído no campo Type é 0x86DD. Os dados são transmitidos para as camadas inferiores usando a primitiva MA-UNITDATA.request.

O processamento de recebimento de um pacote na subcamada LLC funciona da seguinte maneira:

1. A subcamada MAC envia um quadro via primitiva MA-UNITDATA.indication.
2. A subcamada LLC lê o campo Type contendo o valor *EtherType* e extrai o pacote.
 - A. Se o valor no campo Type é 0x88DC, a subcamada LLC entrega o pacote para WSMP.
 - B. Se o valor no campo Type é 0X86DD, a subcamada LLC entrega o pacote para IPv6.
3. Em ambos protocolos, o pacote é entregue da subcamada LLC para as camadas superiores através da primitiva DL-UNITDATA.indication.

3.9 Camadas de Rede e Transporte

Como as camadas de rede e transporte no modelo WAVE estão intrinsecamente relacionadas, especialmente no protocolo WSMP, estas serão discutidas em seção única.

Os protocolos da camadas de Rede e Transporte WAVE são definidos em grande parte no padrão IEEE 1609.3 - Serviços de Rede. Este padrão especifica duas pilhas de protocolos na camada de rede ([Figura 17](#)), IPv6 e WSMP. Esta pilha de protocolos distingue-se entre as duas camadas superiores através do campo *EtherType* conforme explicado na [seção 3.8](#).

O protocolo WSMP foi projetado para otimizar operações em Redes Veiculares, sendo indicado para serviços que necessitam de troca rápida de mensagens. Também é possível a utilização de tráfego IP para serviços que necessitam de conexão com a Internet para obtenção de dados na nuvem. O protocolo IPv6 gera um volume superior de tráfego na rede, sendo apenas permitido nos canais de serviço para não comprometer a performance do WSMP nos canais de controle.

3.9.1 Protocolo WSMP

A família de padrões WAVE é constituída por uma pilha de protocolos de comunicação otimizada para o ambiente veicular, provendo protocolos da camada de rede e

transporte. O protocolo WSMP consiste de uma camada de serviços de rede, uma camada de transporte e uma entidade associada do plano de gerenciamento, a entidade de gerenciamento WAVE (WME ou WAVE Management Entity). O escopo do protocolo WSMP nesta seção é mostrado na [Figura 28](#).

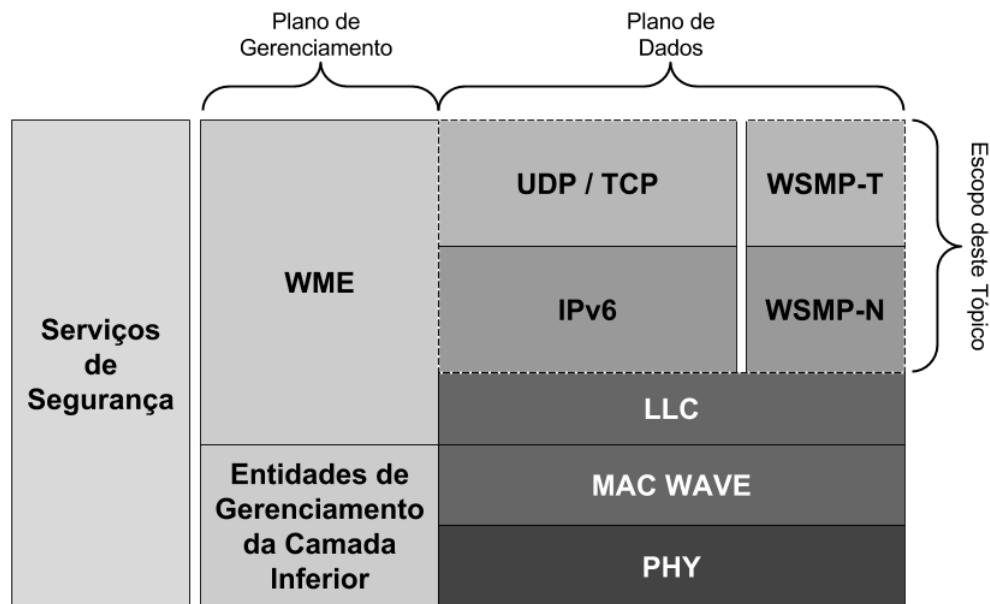


Figura 28 – Escopo do WSMP nesta seção

Fonte: Autor

Nas redes sem fio tradicionais, as aplicações estão preocupadas em apenas enviar e receber informações. Desta forma, detalhes da camada física como canal de transmissão e potência do sinal não são definidos pelas aplicações.

No padrão WAVE, o protocolo WSMP permite que aplicações controlem diretamente as características da camada física (como o canal ou potência do transmissor) a serem utilizadas na transmissão das mensagens. A aplicação de origem também provê um identificador do Provedor de Serviço (PSID ou *(Provider Service Identifier)*), que é utilizado como um direcionador da aplicação no receptor, além de marcar o endereço MAC do dispositivo de destino com a possibilidade de um endereço de grupo para *multicast* ou *broadcast*.

As mensagens curtas WAVE (WSMs ou *WAVE Short Messages*) são entregues para a entidade de destino correta baseados no PSID. Caso o valor do PSID no cabeçalho de uma mensagem recebida representar um serviço que não seja de interesse local, a mensagem pode ser ignorada. Os procedimentos de aceitação de uma mensagem curta WAVE através do PSID serão explicados em maior detalhe na [subseção 3.9.1.3](#).

3.9.1.1 Mensagem Curta WAVE

Quando a camada de aplicação requisita o envio de uma mensagem, esta é encapsulada dentro do campo **dados WSM** de uma mensagem curta WAVE (WSM). Mensagens Curtas WAVE são formadas pelo **cabeçalho** do protocolo WSMP em conjunto com os **dados da WSM**.

A [Figura 29](#) ilustra como este cabeçalho é construído conforme os dados são movidos da camada de aplicação através dos Serviços de Rede³ até a camada MAC. As caixas com borda pontilhada são opcionais e não fazem parte do pacote em si, mas podem ser enviadas para as camadas inferiores como parâmetros através das primitivas ("funções" em itálico na [Figura 29](#)). As caixas com borda preenchida indicam os campos formatados para a transmissão.

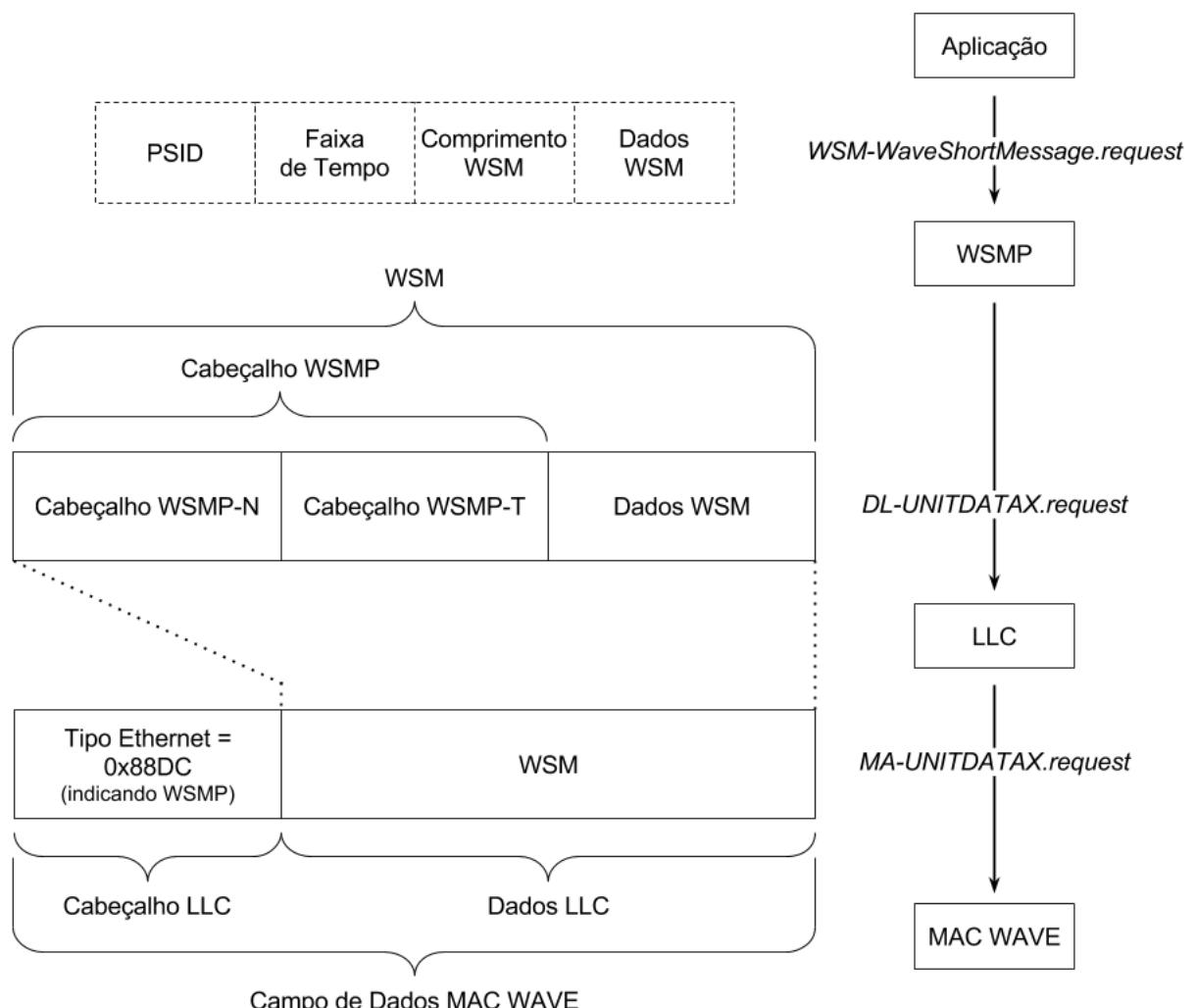


Figura 29 – Composição de uma WSM

Fonte: Autor

³ **Serviços de Rede** compõe os serviços da Camada 3 (Rede) e Camada 4 (Transporte) do Modelo OSI conforme pode ser visto na [Figura 28](#).

O cabeçalho de parte de rede do protocolo WSMP (**WSMP-N**) é composto por um campo **subtipo** de 4 bits, um **indicador de campos opcionais WSMP-N** de 1 bit, um campo de **versão WSMP** de 3 bits, e um campo de identificação para o protocolo de transporte (**TPID**) de 8 bits. Existe também o campo opcional chamado de **informações extensíveis do elemento WAVE**.



Figura 30 – Cabeçalho de Rede WSMP Padrão(Subtipo = 0)

Fonte: Autor

O suporte para o campo **subtipo** com valor 0 (Protocolo de Redes Nulo) é obrigatório. Já o suporte para outros valores de subtipos são possíveis mas não foram descritos no padrão IEEE 1609.3.

O campo **indicador de campos opcionais WSMP-N** contendo valor 1 aponta a presença de **informações extensíveis do elemento WAVE**. Caso o valor seja 0, o campo extensível deve ser desconsiderado. Informações adicionais sobre os campos extensíveis WAVE podem ser encontradas no [Apêndice D](#)).

O campo **versão WSMP** indica qual versão do protocolo WSMP é utilizada. Mensagens curtas WAVE com diferentes valores de versão indicam incompatibilidade entre o formato e o emprego das mensagens. Caso um dispositivo WAVE receba uma mensagem curta com o um valor de versão não suportado, este não processará a mensagem.

O campo **TPID** recebe valores entre 0 e 255. Este campo indica a característica do protocolo de transporte a ser utilizada e especifica qual será a estrutura do cabeçalho de transporte WSMP (**WSMP-T**).

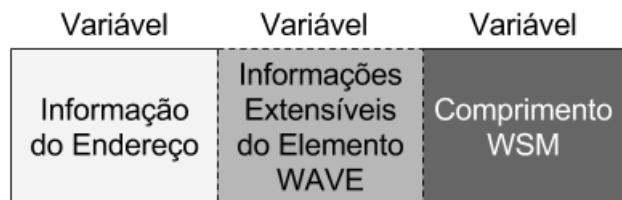


Figura 31 – Cabeçalho de Transporte WSMP Padrão

Fonte: Autor

A Figura 31 demonstra um cabeçalho **WSMP-T** genérico. O conteúdo do campo **informação do endereço** depende do valor do **TPID** inserido. A presença dos campos de **informações extensíveis do elemento WAVE** também depende do **TPID**. O campo **comprimento WSM** é obrigatório e indica o tamanho do campo **dados WSM**.

A Figura 32 apresenta os diferentes formatos do cabeçalho de transporte WSMP (**WSMP-T**). Para valores de **TPIDs** iguais a 1 ou 3, o campo **informações extensíveis do elemento WAVE** estará presente, enquanto nos valores 0 ou 2 não. O **comprimento WSM** sempre estará presente. O campo **informação do endereço** também muda de acordo com o **TPID**.

O suporte para **TPID** 0 e 1 são obrigatórios. Os demais **TPIDs** não são especificados pelo padrão IEEE 1609.3.

Para **TPIDs** com valores iguais a 0 ou 1, a **informação do endereço** é composta apenas do **PSID**. O **PSID** é o endereço de destino que indica qual camada de aplicação deve receber a mensagem. O endereço de origem não é conhecido.

Para **TPIDs** com valores iguais a 2 ou 3, a **informação do endereço** é composta pelos valores das portas de origem e de destino dos dispositivos WAVE.

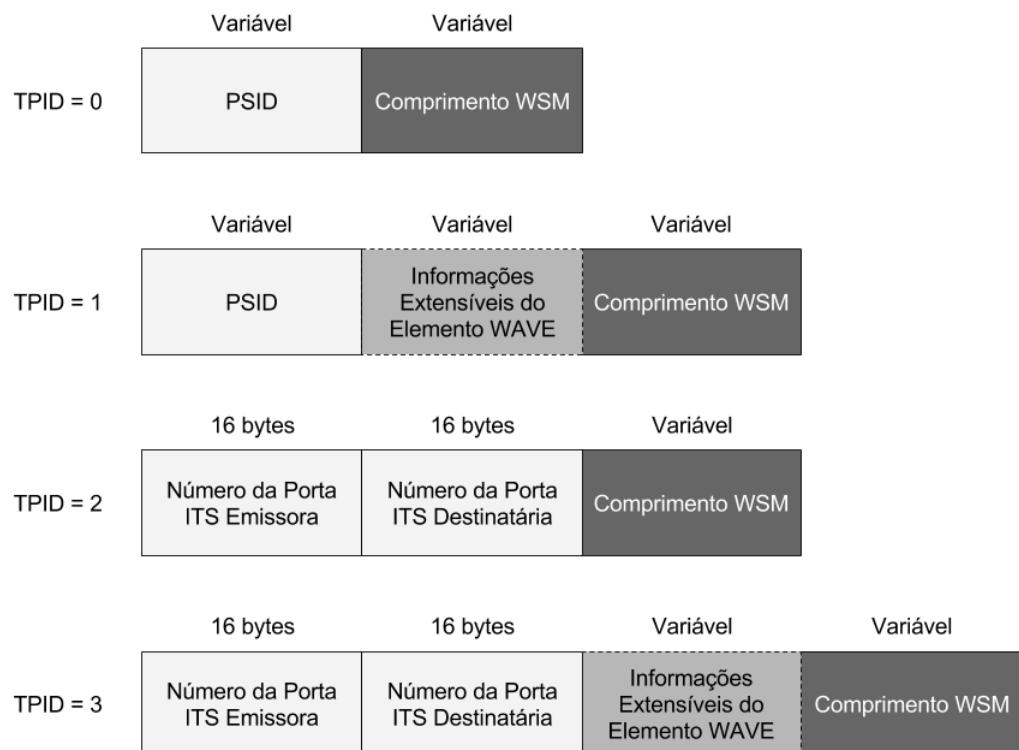


Figura 32 – Diferentes formatos do Cabeçalho WSMP-T de acordo com TPID

Fonte: Autor

3.9.1.2 Mensagem de Anúncio WAVE

Um anúncio de serviço WAVE (WSA) é a compressão das informações de um serviço na forma de mensagens. Conforme a [Figura 33](#), uma WSA formatada (denominada *Ieee1609Dot2Data* pelo padrão WAVE) é inserida no campo **dados WSM** e enviada pelo dispositivo WAVE com o valor do **PSID** 0x87, atribuído para anúncios de serviços WAVE.

Esta mensagem formatada poderá ser uma WSA segura ou não segura.

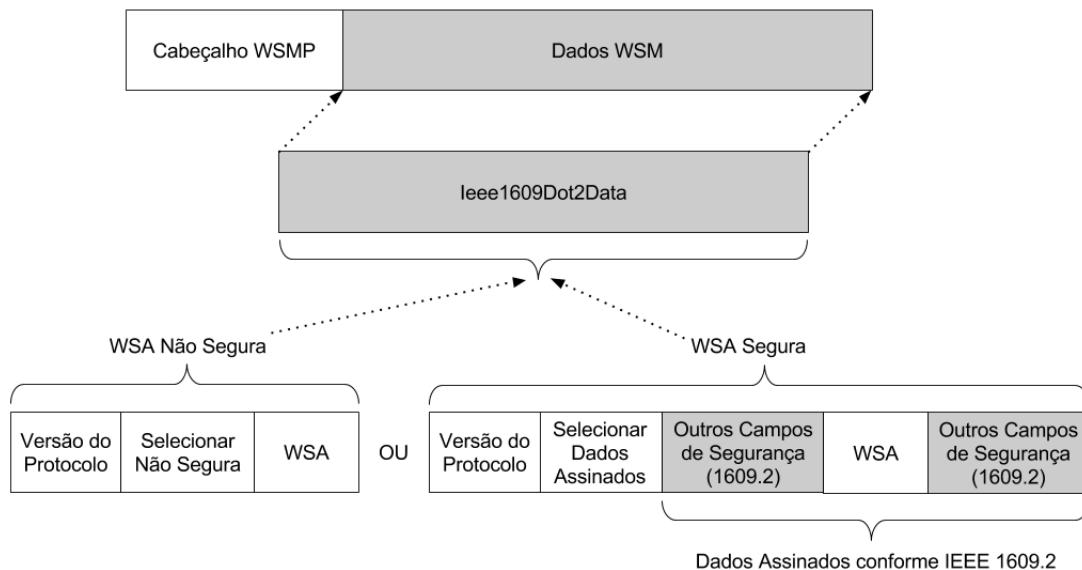


Figura 33 – Modelos de uma WSA

Fonte: Autor

Não há nenhuma especificação para mensagens WSA não seguras, enquanto WSA seguras podem atender a duas especificações diferentes: mensagens assinadas ou mensagens encriptadas.

O padrão IEEE 1609.11 propõe uma aplicação de coleta eletrônica de taxas, onde as trocas de informações para o pagamento de pedágios devem ser encriptadas com intuito de garantir a privacidade dos dados do cartão do motorista. Já mensagens assinadas são utilizadas para garantir que determinado serviço anunciado pertença a um provedor confiável. Conforme proposto por Whyte et al(2015) toda anúncio de serviço WAVE deveria ser assinado, garantindo que punições sejam aplicadas aos provedores que divulgam WSAs maliciosas.

Mensagens assinadas podem contemplar os serviços de: autenticidade, indicando que a mensagem realmente foi enviada pela origem indicada; autorização, assegurando que o dispositivo de origem tem acesso aos privilégios requisitados; e integridade, certificando que qualquer mudança na mensagem será detectável. Mensagens Encriptadas devem ser confidenciais, garantindo que apenas os dispositivos determinados terão acesso

ao conteúdo da mensagem.

É possível configurar os dispositivos WAVE para receberem apenas anúncios de serviços assinados, de forma similar ao que acontece com a escolha de **PSIDs** aceitos pelo dispositivo.

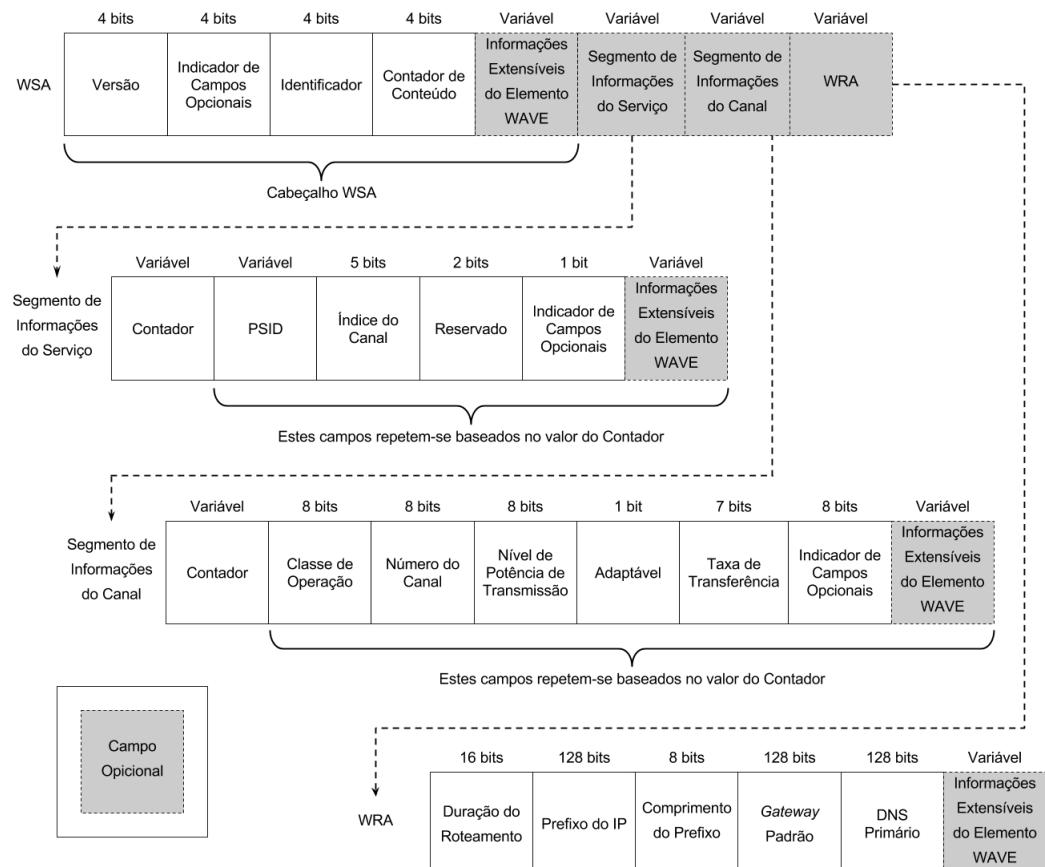


Figura 34 – Cabeçalho de WSA

Fonte: Autor

A Figura 34 exibe o cabeçalho de uma anúncio de serviço WAVE. Pode-se perceber a existência de diversos campos, entretanto, não é de escopo deste trabalho entrar em detalhes nos informações que compõe uma WSA.

3.9.1.3 Transmissão e recepção de mensagens curtas WAVE

Visando uma melhor compreensão da transmissão e recepção de uma mensagem curta WAVE, considere o seguinte exemplo de uma troca de WSMs. Uma aplicação de origem prepara os dados da mensagem para transmissão e seleciona um endereço MAC de broadcast. Baseado nas configurações especificadas para esta mensagem, a aplicação informa as configurações da cama física (potência de transmissão, taxa de transferência) e invoca a primitiva de requisição de mensagem curta WAVE (WSM-WaveShortMessage.request)

para solicitar que o protocolo WSMP entregue o pacote resultante para as camadas inferiores com o intuito de serem transmitidas no canal selecionado.

É possível perceber no exemplo que o canal, a faixa de tempo, a potência de transmissão, a taxa de transferência e o canal podem ser definidos conforme o envio de mensagens são requisitados.

O protocolo WSMP provê funções da camada de rede através do cabeçalho de rede (**WSMP-N**). Este protocolo também fornece funções da camada de transporte através do cabeçalho de transporte (**WSMP-T**). Diferentes primitivas são utilizadas para a troca de informação entre os componentes conforme apresentado na [Figura 35](#).

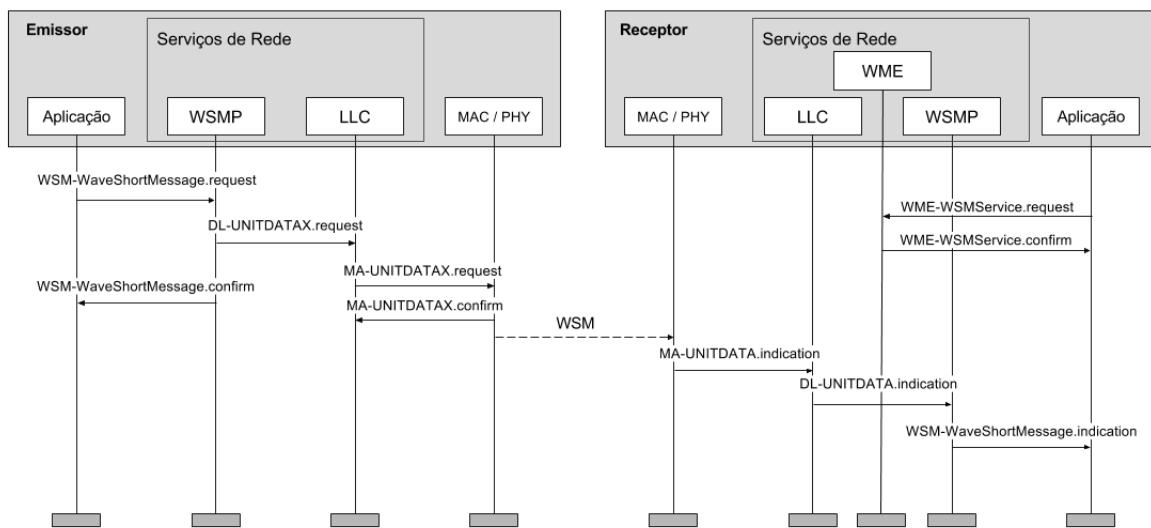


Figura 35 – Transmissão e Recepção de uma WSM

Fonte: Autor

Ao receber uma *WSM-WaveShortMessage.request* de uma aplicação, a camada WSMP⁴ verifica se o comprimento do cabeçalho está de acordo com o limite proposto pelo *WsmMaxLength* da MIB da entidade de gerenciamento WAVE. Caso a verificação seja positiva, o cabeçalho WSMP é gerado e incluído na primitiva *DL-UNITDATA.request*. Caso haja inconsistência na verificação do *WsmMaxLength*, a mensagem curta WAVE não é enviada e a falha é indicada através da primitiva *WSM-WaveShortMessage.confirm*.

A mensagem curta WAVE recebida na camada física é entregue para a aplicação especificada no campo **PSID**. O dispositivo receptor aceita a mensagem e a envia para as camadas superiores na pilha de protocolos ([Figura 28](#)). A camada MAC verifica se a mensagem é destinada à ele (*Unicast*, *Multicast* ou *Broadcast*). Caso a verificação seja positiva, o **PSID** é verificado pela camada WSMP para saber se há interesse neste tipo de mensagem. Caso haja o interesse, a camada MAC envia esta mensagem para as camadas

⁴ Serviços de Rede e Transporte vide [Figura 28](#)

superiores. A mensagem recebida será enviada pela camada WSMP para quaisquer entidades da camada de aplicação com interesse no serviço associado àquele **PSID**. O interesse é indicado através do *WME-WSMService.request*.

Ao receber uma *DL-UNITDATA.indication* da subcamada LLC, a camada WSMP avalia a **versão WSMP**. Se o dispositivo WAVE receber uma mensagem curta com uma **versão WSMP** não suportada, nenhum processamento adicional é necessário. Em seguida, a camada WSMP avalia o **subtipo** e processa o **TPID**. As entidades da camada superior são notificados da recepção da WSM através da primitiva *WSM-WaveShortMessage.indication*. Após o processamento do cabeçalho, a camada WSMP entrega o valor do campo **dados WSM** para as aplicações com interesse no **PSID** recebido, os destinatários são indicados na tabela MIB *WsmServiceRequestTable*, que é uma tabela estabelecida com cada *WME-WSMService.request* executado. Se um dispositivo WAVE receber uma mensagem curta WAVE onde o **subtipo** for diferente de 0, ou o **TPID** for diferente de 0 e 1, então não há a necessidade de processamento adicional.

3.9.2 Protocolo IPv6

Na camada de rede, o suporte WAVE ao protocolo IPv6 deve obedecer as especificações da norma IETF RFC 2460. O tráfego IP das Redes Veiculares é transmitido e recebido através da subcamada de enlace LLC com o campo *Type* contendo valor *Ethertype* 0x86DD.

Os serviços com tráfego IPv6 nas Redes Veiculares são controlados por unidades de acostamento ligadas à redes de infraestrutura. Como exemplo na [Figura 36](#), um serviço de informação sobre o clima pode operar através de uma RSU. As unidades de bordo WAVE interessadas no serviço e dentro da área de cobertura podem obter endereços IPv6 a fim de operar na rede de infraestrutura, com objetivo de obter informações mais precisas sobre o serviço oferecido. Os endereços da rede são distribuídos pelo dispositivo WAVE localmente e podem ser usados sem necessidade de configuração externa.

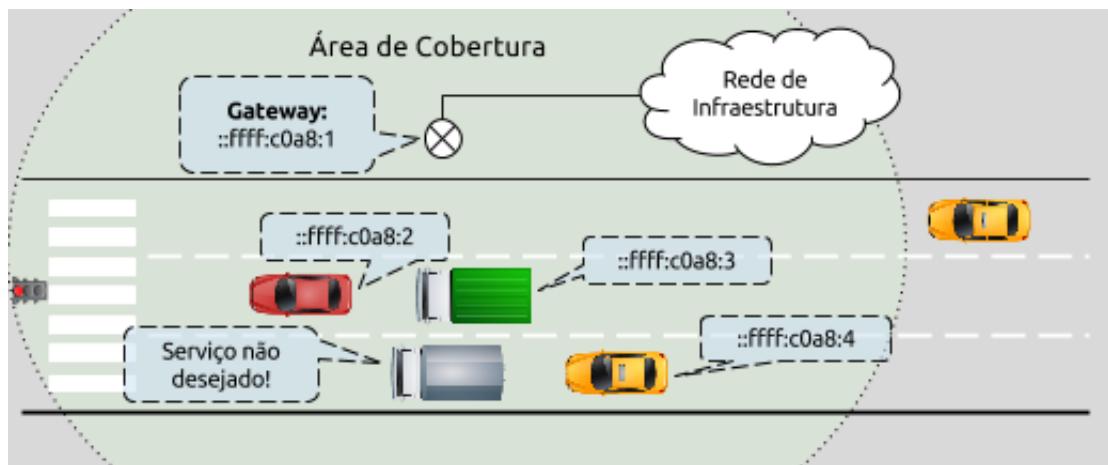


Figura 36 – Atribuição de IPv6 por uma unidade de acostamento

Fonte: Autor

Cada dispositivo WAVE que oferecer serviços IP deverá suportar endereços locais, globais e de *multicast* para IPv6, seguindo a norma IETF RFC 2373.

A configuração IP do padrão WAVE será realizada através de WSAs utilizando um canal de serviço, já que o canal de controle não podem ter tráfego IP.

Os dispositivos WAVE podem implementar qualquer outro protocolo da IETF (Internet Engineering Task Force), mas devem ser tomadas precauções para que a interoperabilidade não seja afetada com outros dispositivos da Rede Veicular.

Portanto, conforme visto nesta seção, a comunicação entre veículos nas camadas de rede e transporte pode ser feita através dos protocolos IPv6 ou WSMP. O protocolo WSMP é utilizado para troca de informações *ad hoc* dentro da rede veicular, já o IPv6 é empregado em situações quando a troca de dados necessita de uma rede de infraestrutura.

3.10 Camada de Aplicação

Utilizando a troca de mensagens na camada de aplicação WAVE, softwares podem ser desenvolvidos para gerar funcionalidades voltadas à segurança, informação e entretenimento nas redes veiculares.

3.10.1 Aplicações de Segurança e Informação

A SAE, através da norma SAE J2735, propôs três diferentes conjuntos de mensagens para aplicações de redes veiculares. Estas mensagens foram propostas de modo pragmático, utilizando notação ASN.1 e regras de codificação nos formatos DER (*Distinguished Encoding Rules*) e XML. É importante salientar que o padrão SAE J2735 não

implementa uma aplicação ou sistema e sim ilustra como as mensagens da camada de aplicação são codificadas e decodificadas.

Como até então não haviam APIs (*Application Program Interface*) desenvolvidas para as camadas do modelo WAVE e nem protótipos de placas de rede, o padrão realizou testes de construção e decodificação de mensagens broadcast WSM utilizando *sockets* Berkeley como API padrão, além do protocolo de transporte UDP representando o WSMP em uma placa de rede *Ethernet*. Entidades de gerenciamento WAVE foram criadas no código.

Os conjuntos de mensagens implementados no padrão SAE J2735 são:

- BSM (*Basic Safety Message* ou Mensagem Básica de Segurança)
- RSA (*RoadSide Alert* ou Mensagens de Alerta de Acostamento)
- Mensagens Sensoriais de Veículos (*Probe Vehicle Message* ou PVM)

As BSMs são mensagens para aplicações de segurança em veículos. Estas aplicações utilizariam o fluxo de dados de BSMs provenientes de veículos nas redondezas para detectar potenciais eventos de risco. As BSMs fornecem aos veículos conhecimento da posição, velocidade e direção dos veículos nas imediações. Estas mensagens de segurança possuem grupos de PSIDs para dados específicos. Conforme mostrado [Tabela 19](#), para BSMs contendo dados de alta precisão o grupo PSID utilizado é o 0x20. Para veículos com dados não tão acurados, o grupo de PSID utilizado é o 0x21.

Como exemplo, foi proposta uma aplicação de um sistema de frenagem inteligente que detecta potenciais colisões traseiras através de algoritmos de detecção. Após detectado o potencial risco de colisão, o condutor é alertado passivamente ou o sistema realiza ações corretivas, como desabilitar o “piloto automático”, fortalecer os freios e pré-tensionar os cintos de segurança.

Outros exemplos de aplicações podem aproveitar-se das mensagens básicas de segurança para os mais diversos fins, como:

- Aplicações para aviso de pontos cegos;
- Aplicações para cooperação entre pilotos automáticos;
- Aplicações para prevenção de colisões;
- Aplicações para advertir presença de veículos de emergência;
- Aplicações para alertar mudança de faixa;
- Aplicações para detecção pré-colisão

As RSAs são mensagens de broadcast voltadas à aplicações de informação. Estas informações são voltadas principalmente às condições de tráfego e são propagadas principalmente por veículos de segurança pública ou por unidades de acostamento (RSUs ou *RoadSide Units*) que entregam dados de Centros de Gerenciamento de Tráfego. Os tipos de dados transmitidos por RSAs são geralmente de atrasos em percursos causado por engarrafamentos, recálculo de rotas, informações de sinalização, incidentes, pontos de interesse, etc.

As PVMs são mensagens usadas por múltiplas aplicações. Os veículos coletam dados próprios, de condições de pistas e de tráfego através de seus sensores em determinados intervalos de tempo. Esta gama de dados coletada é chamada de *snapshot*. Estes dados são combinados em mensagens e transmitidos para unidades de acostamento (RSUs) que aceitam estes *snapshots* quando dentro de seu alcance. Das RSUs, os dados são enviados para Centros de Gerenciamento de Tráfego e Provedores de Serviços a fim de outros usuários móveis serem abastecidos destas informações.

Como exemplo, os sensores de dados retornam informações como:

- Aceleração Vertical e Horizontal
- Altitude
- Ângulo do volante
- Calibragem dos pneus
- Clima (Sol, Chuva, Radiação)
- Coeficiente de Fricção com a pista
- Direção de obstáculo
- Direção do veículo
- Distância de obstáculo
- Latitude
- Longitude
- Pressão atmosférica
- Status do Sistema de Freio
- Taxa de derrapagem
- Temperatura

- Tempo e hora
- Tipo de veículo
- Velocidade

3.10.2 Coleta eletrônica de Taxas

Baseando-se nos padrões IEEE 1609.3, IEEE 1609.4, IEEE 1609.11, ISO 14906 e ISO 15628, um sistema de pagamento eletrônico para redes veiculares foi proposto como aplicação em redes veiculares. As normas citadas estabelecem as diretrizes para alcance de confiabilidade e segurança no pagamento.

Para transações eletrônicas ocorrerem com sucesso, deve-se garantir uma associação inequívoca de cada pagador com seu veículo para que não haja erros no processamento do pagamento durante a curta presença dos veículos ao atravessarem os postos de pagamento eletrônicos (que podem ser de estacionamentos ou pedágios utilizando-se ou não de cancelas eletrônicas). Isto é, se cinco veículos passarem por um posto eletrônico de pedágio e apenas 4 pagamentos forem registrados, o sistema deve determinar qual veículo evadiu o pagamento. Fotografias dos veículos em conjunto com o registro de transações realizadas podem evidenciar possíveis evasões.

A Figura 37 apresenta uma aplicação de um sistema de pagamento eletrônico para pedágios. Com duas unidades de acostamento (RSUs) de três antenas na praça de pedágio, a primeira antena de cada unidade é sintonizada no Canal de Controle (CCH) para enviar frames “despertadores” para as unidades móveis próximas.

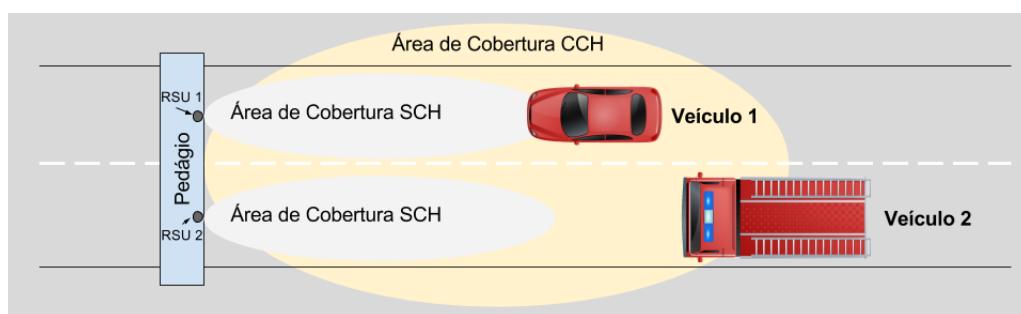


Figura 37 – Aplicação de pedágio

Fonte: Autor

A antena no CCH também envia Mensagens de Anúncio que divulgam a presença da praça de pedágio no percurso e entregam os parâmetros do serviço como o canal, método de acesso ao canal, PSID e demais parâmetros.

Quando uma unidade móvel (OBU) recebe o anúncio de um Serviço de Coleta Eletrônica de Taxas, a entidade de gerenciamento WAVE (WME) troca para o canal de

serviço (SCH) e método de acesso descritos pelo anúncio, que no exemplo é respectivamente SCH 182 com acesso contínuo. Informações da aplicação na [Tabela 4](#).

As duas antenas restantes se responsabilizam pelo SCH a fim de executar a troca de mensagens com OBUs na área de cobertura. Logo, as camadas de aplicação da RSU e das OBUs dentro da área de cobertura trocam informações de pagamento.

Parâmetro	Valor	Descrição
Endereço MAC Destinatário	broadcast	WSA é um broadcast
Tipo WSA	não segura	WSA não assinada
PSID	0x01	Coleta eletrônica de taxas
Número do Canal	182	Identifica o número do canal a ser utilizado pelo serviço
Tipo de Acesso ao Canal	contínuo	Usuário deve trocar para SCH imediatamente
Taxa de repetição	250	WSA transmitida 5 vezes a cada 100ms
Serviço de IP	falso	Utiliza-se WSMP

Tabela 4 – Dados da aplicação de pedágio

É importante acrescentar que a realização do pagamento eletrônico leva em consideração que os veículos têm algum mecanismo de pagamento cadastrado e funcional.

O próximo capítulo irá discorrer sobre as diversas formas de avaliar novas tecnologias.

4 Abordagem Avaliativas

Toda e qualquer tecnologia precisa ser testada utilizando abordagens avaliativas antes de entrar em caráter de produção no mercado. Abordagens avaliativas são métodos distintos de pensar, desenhar e conduzir atividades de avaliação. A utilização de abordagens avaliativas na ciência cresceu consideravelmente a partir de um movimento internacional encorajando práticas científicas baseadas em evidências. A abordagem avaliativa empregada em determinada tecnologia é fundamental para antecipar possíveis inconvenientes ou conflitos antes desta entrar em fase de produção.

Para a implementação de tecnologias em redes veiculares, há um conjunto de abordagens avaliativas que podem ser empregadas que vão desde coletas da análise conjuntos de dados, passando por simulações até experimentos no mundo real.

A utilização de ferramentas de simulação foi a abordagem experimental escolhida por este trabalho para o experimento de redes veiculares em estudo de caso. Banks et al (2005) cita diversos propósitos em que simulações são ferramentas apropriadas de avaliação, entre as quais se destacam para o escopo deste projeto:

- O conhecimento alcançado durante o desenho do módulo de simulação é de valor considerável para a sugestão de aprimoramentos no sistema sob investigação;
- A simulação pode ser usada como instrumento pedagógico para reforçar solução analíticas;
- A simulação pode ser utilizada para experimentar novos conceitos antes da implementação, como também preparar para situações inesperadas;
- A simulação pode testar diferentes recursos para a máquina que podem auxiliar em uma futura determinação de requisitos.

Banks et al (2005) acrescem que as simulações devem portar dados de saída que correspondam diretamente àqueles gravados em mundo real para que suposições duvidosas não sejam criadas a seu respeito. O processo avaliativo envolvido na simulação é de caráter observativo. Um conjunto de dados de entrada e características é dado, o módulo é executado para então resultar em cenários a serem avaliados.

Pegden et al (1995) citam vantagens no desenvolvimentos de módulos para simulação:

- Novas políticas, regras, fluxos, procedimentos organizacionais e operacionais podem ser explorados sem interromper as operações em andamento no mundo real;

- Novos esboços de hardware podem ser testados sem necessidade de reservar recursos para sua aquisição;
- Hipóteses sobre como certos fenômenos ocorrem podem ser testados;
- O tempo pode ser compresso ou expandido para permitir aceleração ou desaceleração dos fenômenos sob investigação;
- A compreensão das interações, das variáveis e sua importância pode ser aprimorada na observação de simulações;
- A simulação permite analisar gargalos a fim de descobrir onde há processos atrasando o desempenho do sistema;

Confome corroborado também por Olariu & Weigle (2009), a maneira mais apropriada de se avaliar o desempenho de uma rede é desenvolvendo simulações que forneçam resultados mais próximos de observações do mundo real.

Para se avaliar o funcionamento de simulações em redes veiculares, há de se integrar um conjunto de softwares que incorporem o aspecto de arquitetura de redes de comunicações juntamente com softwares que produzam a mobilidade e a aleatoriedade existente em ambientes veiculares. Isto é, para a simulação de redes veiculares é necessário integrar:

- Um software de Simulação de Redes
- Um software de Simulação de Tráfego Urbano
- Um *Middleware* que coordene comunicação bidirecional

Cheng et al (2015) analisaram um considerável número de softwares destes grupos e coletaram características individuais para concluir quais seriam os conjuntos de softwares de simulação de redes veiculares que trariam resultados mais próximos à realidade. No fim de cada seção seguinte, serão listados requisitos importantes para a escolha de um software de cada tipo e anunciado o software escolhido para o estudo de caso deste trabalho.

4.1 Softwares de Simulação de Redes

Os softwares de simulação de redes auxiliam no desenvolvimento de novos projetos de redes. Alguns destes softwares oferecem interface gráfica (*Graphical User Interface* ou *GUI*), além de módulos de tráfego para diversos protocolos, módulos de perda de pacotes, de atenuação de sinal, de tipos de cabos e protocolos de rede desejados. Fabricantes de dispositivos de rede como a Cisco utilizam de softwares de simulação para demonstrar o funcionamento de seus produtos.

Através das simulações desempenhadas em softwares de simulação de redes, o desempenho de diferentes configurações de redes pode ser comparado, sendo possível reconhecer e resolver problemas de desempenho sem a necessidade de conduzir testes de campo que seriam potencialmente caros. A simulação de redes é amplamente utilizada por pesquisadores na avaliação do comportamento de protocolos recém desenvolvidos e por *designers* de rede para desenhar e avaliar novos projetos de rede.

Na maioria dos casos, softwares de simulação de redes são baseados no conceito de modelagem de eventos discretos, que consiste na mudança de estado de sistemas apenas no instante que um determinado evento ocorrer. Para os demais instantes de tempo, nada muda no sistema. Os modelos de simulação utilizam métodos numéricos, isto é, procedimentos computacionais para lidar com modelos matemáticos.

Como redes veiculares utilizam de comunicações sem fio, nem todos os softwares de simulação de redes podem ser utilizados para a construção deste cenário de simulação exceto aqueles que apresentarem módulos de simulação de redes veiculares integrados ou externos.

Cheng et al (2015) analisaram softwares de simulação de redes, distinguindo-os conforme mostrado na [Tabela 5](#).

Simulador de Redes	OPNET	ns-2	ns-3	J-Sim	OMNET	SWANS
Licença	Comercial	GNU GPLv2	GNU GPLv2	BSD License	Acadêmica, Comercial	Acadêmica
Linguagem de Programação escrita	-	C++	C++	Java	C++	Java
Linguagem da simulação	C, C++	C++, OTcl	C++, Python	Java, Tcl, Perl, Python	C++, NED	Java, Python
Possui GUI	Sim	Não	Não	Não	Sim	Não
Sistemas Suportados	Windows	Linux, OS X, Solaris, Windows	Linux, Windows	Windows, OS X, Linux	Windows, OS X, Linux	Windows, OS X, Linux
Código Fonte	Apenas para nós de simulação	Sim	Sim	Sim	Sim	Sim
Tecnologias sem-fio	<i>WiFi</i> , LTE, UMTS, <i>WiMAX</i> , Zigbee, <i>Satellite</i>	<i>WiFi</i> , <i>Satellite</i> , <i>Cellular</i>	<i>WiFi</i> , <i>WiMAX</i> , LTE	<i>WiFi</i>	Nenhuma (Apenas módulos externos)	<i>WiFi</i> (IEEE 802.11b)
Módulo de mobilidade	Não	Não	Não	Não	Não	Não

Tabela 5 – Comparação entre softwares de simulação de redes

Para determinar o software de simulação de redes para o desenvolvimento de estudo de caso, o processo de escolha atendeu os seguintes requisitos:

1. O software possui uma interface gráfica amigável para compreensão ágil de seu escopo de desenvolvimento;
2. O software apresenta portabilidade para diversos sistemas operacionais;
3. O software é de licença acadêmica ou aberta (*opensource*) ;

4. O software permite a simulação modelos de redes veiculares;
5. O software possui boa reputação na campo acadêmico com uma comunidade ativa.

O software de simulação que atendeu melhor tais critérios foi o **OMNET++**. Este software é descrito com maiores detalhes na próxima seção.

4.1.1 OMNET++

O OMNET++ é um software de simulação de eventos discretos que possui licença acadêmica pública. As bibliotecas componentes deste software foram construídas na linguagem C++. Este software apresenta características de modularidade, extensibilidade e portabilidade. A primeira e segunda características dizem respeito à possibilidade de integração com outras bibliotecas, tornando possível simular, por exemplo, redes de sensores, redes *ad hoc*, protocolos de Internet, redes fotônicas. Isto fez do OMNET++ conhecido na comunidade científica como um software de simulação de redes, justamente por conter bibliotecas de simulação de redes como MiXiM, INET e VEINS. A portabilidade do *framework* é vista no suporte à diversas plataformas, como os sistemas Windows, Linux e Mac OS X.

A interface gráfica do software é baseada no Eclipse, o que torna a ambientação mais rápida para desenvolvedores acostumados a trabalhar com esta IDE. A [Figura 38](#) apresenta a IDE do OMNET++.

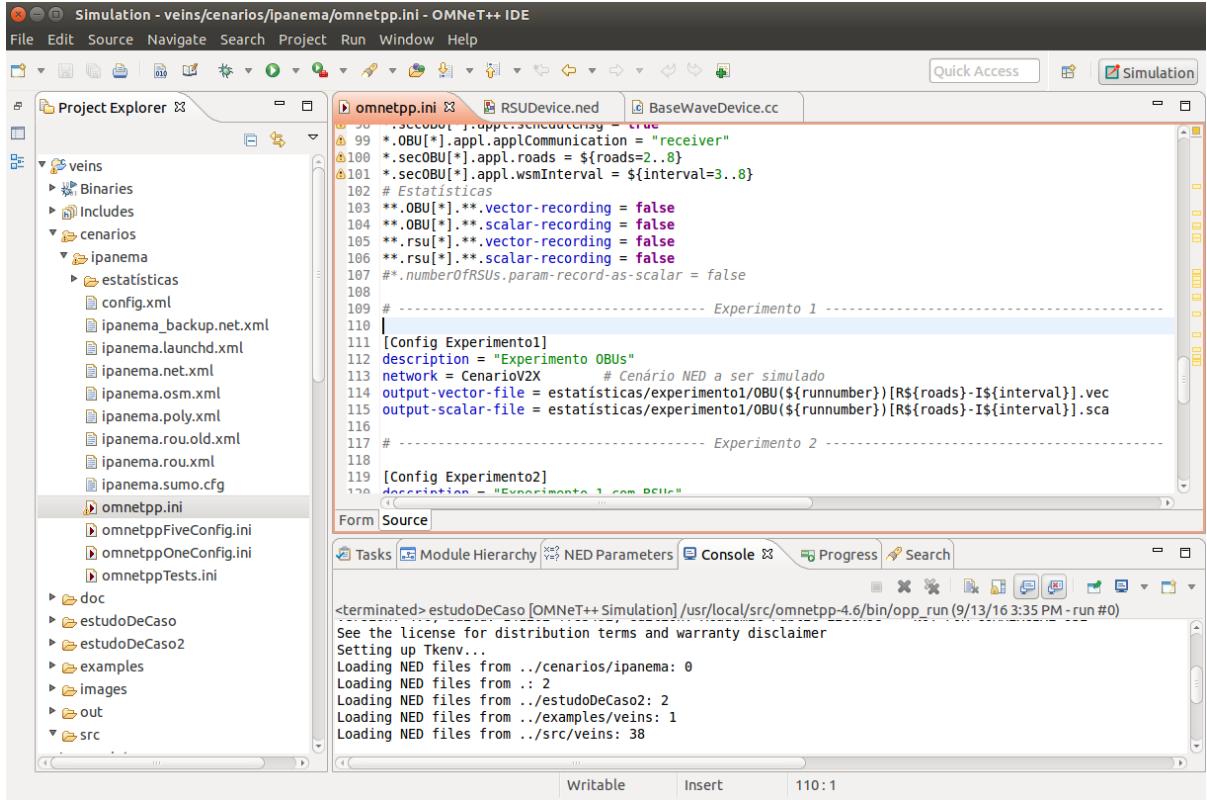


Figura 38 – Interface OMNET++

Fonte: omnetpp.org

O software possui três ambientes de execução, sendo duas gráficas. O Cmdenv, que executa as simulações em lote pela linha de comando. O TkEnv, que é o mais tradicional e permite a virtualização de objetos em duas dimensões ([Figura 39](#)). E por fim, o recém incorporado QtEnv permite a simulação gráfica de objetos em três dimensões, utilizando a biblioteca do *openscenegraph* ([Figura 40](#)).

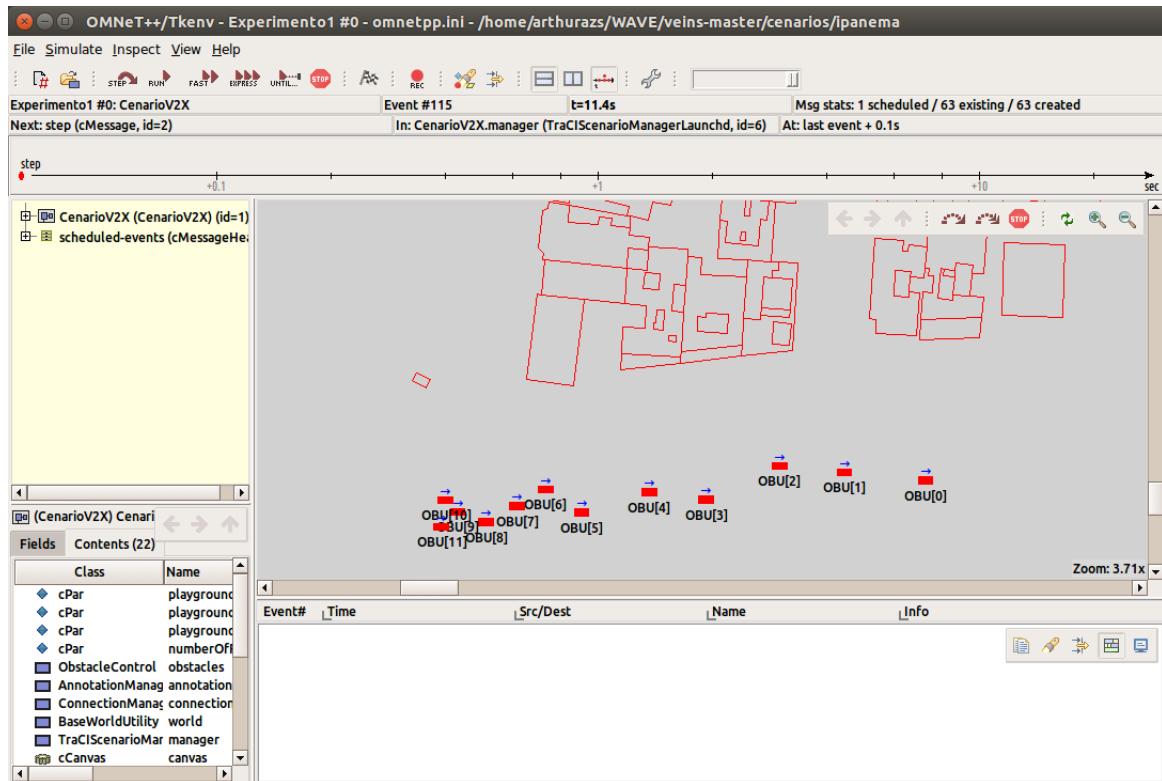


Figura 39 – Interface TkEnv

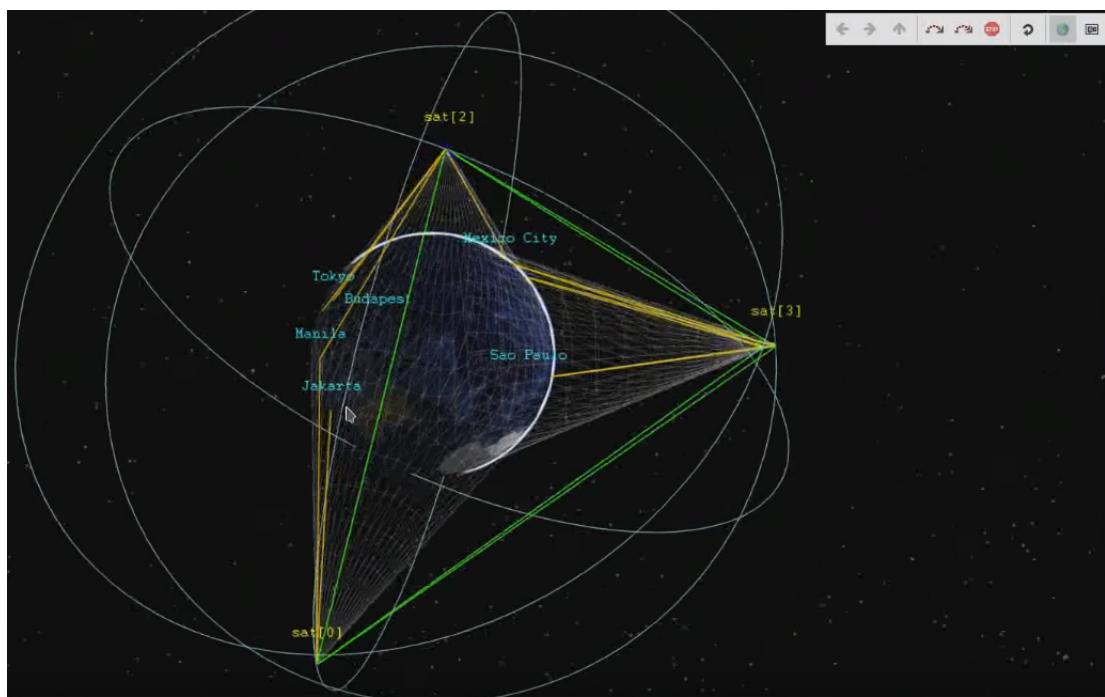
Fonte: omnetpp.org

Figura 40 – Interface QtEnv

Fonte: omnetpp.org

Além de realizarem a simulação gráfica, o TkEnv e QtEnv fazem o registro de eventos. Há GUIs específicas para os registros dos módulos (Figura 41) e de pacotes enviados entre os nós (Figura 42).

```

INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Channel is already idle for:3.999873997466 since 108.103701949359
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Waiting for Queue 2:0.000071=3 * 0.000013 + 0.000032*0*0.000013; Idl
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Could have already send if we had it earlier
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Updated nextMacEvent:112103586949359
** Event #2173 t=112.103586949359 CenarioV2X.emergencyOBU[0].nic.mac1609_4 (Mac1609_4, id=12) on selfmsg next Mac Event (omnetpp::cMessage, id=1541)
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Channel turned busy: Switch or Self-Send
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Stopping Contention at 112103586949359
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Channel was idle for 3.999885
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Slotted packets after DIFS: 307678
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Updating backoff for Queue 2: 0 -> 0 TXOP
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Initiating transmit at 112.103586949359. I've been idle since 3.999885
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Queue 2 is ready to send!
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: MacEvent received. Trying to send packet with priority2
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Sending duration will be0.000121
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: Sending a Packet. Frequency 5.89e+0 Priority2
** Event #2174 t=112.103586949359 CenarioV2X.emergencyOBU[0].nic.mac1609_4 (Mac1609_4, id=12) on Radio switching over (omnetpp::cMessage, id=2807)
INFO (Mac1609_4)CenarioV2X.emergencyOBU[0].nic.mac1609_4: PhyLayer said radio switching is done
** Event #2175 t=112.103587949359 CenarioV2X.emergencyOBU[0].nic.phy80211p (PhyLayer80211p, id=29) on data (AirFrame1ip, id=2805)
INFO (PhyLayer80211p)CenarioV2X.emergencyOBU[0].nic.phy80211p: emergencyOBU[0]:PhyLayer80211p: AirFrame encapsulated, length: 1326
** Event #2176 t=112.10358832052 CenarioV2X.basicOBU[3].nic.phy80211p (PhyLayer80211p, id=29) on data (AirFrame1ip, id=2805)
INFO (PhyLayer80211p)CenarioV2X.basicOBU[3].nic.phy80211p: basicOBU[3]:PhyLayer80211p: Received new AirFrame (AirFrame1ip) data from channel.
INFO (PhyLayer80211p)CenarioV2X.basicOBU[3].nic.phy80211p: PhyLayer(SimplePathlossModel): srdistance is: 70026.2
INFO (PhyLayer80211p)CenarioV2X.basicOBU[3].nic.phy80211p: PhyLayer(SimplePathlossModel): wavelength is: 0.0509985
INFO (PhyLayer80211p)CenarioV2X.basicOBU[3].nic.phy80211p: PhyLayer(SimplePathlossModel): distance factor is: 9.04315e-08
INFO (PhyLayer80211p)CenarioV2X.basicOBU[3].nic.phy80211p: PhyLayer(SimplePathlossModel): Signal contains frequency dimension: yes
INFO (PhyLayer80211p)CenarioV2X.basicOBU[3].nic.phy80211p: PhyLayer(SimpleObstacleShadowing): value is: 1.09904e-06
INFO (PhyLayer80211p)CenarioV2X.basicOBU[3].nic.phy80211p: [Host 3] - PhyLayer(Decider): Processing AirFrame...
INFO (PhyLayer80211p)CenarioV2X.basicOBU[3].nic.phy80211p: 1.00051e-11 > 3.16228e-07 = 0
INFO (PhyLayer80211p)CenarioV2X.basicOBU[3].nic.phy80211p: basicOBU[3]:PhyLayer80211p: Handled AirFrame with ID 45 to Decider. Next handling in 0.00012s.

```

Figura 41 – Module Logger

Fonte: omnetpp.org

Event#	Time	Src/Dest	Name	Info
#1758	92.108529041731	basicOBU[2] --> emergencyOBU[0]	data	id=2579 kind=22003 length=166 bytes
#1758	92.108529041731	basicOBU[2] --> basicOBU[1]	data	id=2581 kind=22003 length=166 bytes
#1758	92.108529041731	basicOBU[2] --> basicOBU[3]	data	id=2578 kind=22003 length=166 bytes
#1833	96.103582272855	emergencyOBU[0] --> basicOBU[1]	data	id=2614 kind=22003 length=166 bytes
#1833	96.103582272855	emergencyOBU[0] --> basicOBU[2]	data	id=2615 kind=22003 length=166 bytes
#1833	96.103582272855	emergencyOBU[0] --> basicOBU[3]	data	id=2613 kind=22003 length=166 bytes
#1855	96.106556611104	basicOBU[3] --> emergencyOBU[0]	data	id=2651 kind=22003 length=166 bytes
#1855	96.106556611104	basicOBU[3] --> basicOBU[1]	data	id=2653 kind=22003 length=166 bytes
#1855	96.106556611104	basicOBU[3] --> basicOBU[2]	data	id=2650 kind=22003 length=166 bytes
#1877	96.108529616189	basicOBU[2] --> emergencyOBU[0]	data	id=2684 kind=22003 length=166 bytes
#1877	96.108529616189	basicOBU[2] --> basicOBU[1]	data	id=2686 kind=22003 length=166 bytes
#1877	96.108529616189	basicOBU[2] --> basicOBU[3]	data	id=2683 kind=22003 length=166 bytes
#1983	96.111776999061	basicOBU[1] --> emergencyOBU[0]	data	id=2727 kind=22003 length=166 bytes
#1983	96.111776999061	basicOBU[1] --> basicOBU[2]	data	id=2729 kind=22003 length=166 bytes
#1983	96.111776999061	basicOBU[1] --> basicOBU[3]	data	id=2726 kind=22003 length=166 bytes
#1974	100.103582949359	emergencyOBU[0] --> basicOBU[1]	data	id=2754 kind=22003 length=166 bytes
#1974	100.103582949359	emergencyOBU[0] --> basicOBU[2]	data	id=2756 kind=22003 length=166 bytes
#1974	100.103582949359	emergencyOBU[0] --> basicOBU[3]	data	id=2753 kind=22003 length=166 bytes
#2041	104.103588949359	emergencyOBU[0] --> basicOBU[1]	data	id=2773 kind=22003 length=166 bytes
#2041	104.103588949359	emergencyOBU[0] --> basicOBU[2]	data	id=2775 kind=22003 length=166 bytes
#2041	104.103588949359	emergencyOBU[0] --> basicOBU[3]	data	id=2772 kind=22003 length=166 bytes
#2108	108.103581949359	emergencyOBU[0] --> basicOBU[1]	data	id=2792 kind=22003 length=166 bytes
#2108	108.103581949359	emergencyOBU[0] --> basicOBU[2]	data	id=2794 kind=22003 length=166 bytes
#2108	108.103581949359	emergencyOBU[0] --> basicOBU[3]	data	id=2791 kind=22003 length=166 bytes
#2175	112.103587949359	emergencyOBU[0] --> basicOBU[1]	data	id=2811 kind=22003 length=166 bytes
#2175	112.103587949359	emergencyOBU[0] --> basicOBU[2]	data	id=2813 kind=22003 length=166 bytes
#2175	112.103587949359	emergencyOBU[0] --> basicOBU[3]	data	id=2810 kind=22003 length=166 bytes

Figura 42 – Message Logger

Fonte: omnetpp.org

O OMNET++ fornece uma arquitetura de componentes para os módulos a serem simulados. Este componentes são programados em C++, e integrados em uma linguagem de alto nível chamada NED. O principal objetivo da linguagem NED é descrever as topologias de rede a serem utilizadas. Os módulos descritos podem ser estendidos ou reutilizados por qualquer outro externo. A GUI do OMNET++ também pode ser estendida.

4.2 Software de Simulação de Tráfego

Os softwares de Simulação de Tráfego são utilizados na Engenharia de Transportes e ajudam a descrever o desenho do trânsito de uma cidade da forma mais eficiente possível. Estes *softwares* utilizam modelos de tráfego que se classificam em três tipos no que diz respeito à granularidade do fluxo:

Modelos Macroscópicos medem o tráfego em larga escala, tratando o tráfego como um líquido. O nível de detalhamento do fluxo de tráfego nestes modelos é baixo, não existindo interesse por cada veículo e sim pelo fluxo.

Modelos Mesoscópicos medem o tráfego em uma escala menor, se preocupando com o movimento de pelotões de veículos. Utilizam funções velocidade-densidade para modelar o comportamento destes modelos.

Modelos Microscópicos medem o tráfego de veículos em uma escala individual em que cada veículos é um traço, portanto o nível de detalhamento do fluxo de tráfego nestes modelos é alto.

Granularidade do Modelo de Tráfego	Frameworks
Macroscópica	METACOR
Mesoscópica	CONTRAM
Microscópica	SUMO

Tabela 6 – Modelos de tráfego por granularidade

Para a simulação de redes veiculares, foi notado que o modelo de tráfego deveria se preocupar com a modelagem individual de transmissão de rádio entre os nós, necessitando das posições, dos vetores e de ações exatas de nós individuais. Modelos com pelotões de veículos e ou tendências macroscópicas de tráfego não se adequariam às comunicações inter veiculares.

Seguindo o critério estabelecido acima foi visto que há diversos modelos de microtráfego urbano implementados. Segundo conclusão de Brockfeld & Wagner (2004), todos os modelos microscópicos de simulação apresentam um mesmo valor de acurácia como modelos de mobilidade. Isto inclui os modelos *Cellular Automaton*, *Car-following* e *IDM/MOBIL*.

O processo de escolha de software de simulação de trânsito para o desenvolvimento de estudo de caso atendeu os seguintes critérios:

1. O software apresenta um modelo de trânsito com granularidade microscópica;
2. O software apresenta portabilidade para diversos sistemas operacionais;

3. O software é de licença acadêmica ou aberta (*opensource*);
4. O software permite a simulação modelos de redes veiculares;
5. O software possui boa reputação na campo acadêmico com uma comunidade ativa.

O software de simulação que cumpriu estes critérios foi o **SUMO** (Simulator of Urban Mobility). Este software é descrito em maior detalhe na próxima seção.

4.2.1 Simulator of Urban Mobility

O SUMO é um software de simulação de tráfego urbano desenvolvido pelo Instituto de Sistemas de Transportes da Alemanha. É um sistema de código aberto (*opensource*), portátil e foi projetado para lidar com grandes redes rodoviárias.

O SUMO faz a modelagem de tráfego microscópica. Cada veículo tem sua própria rota e se move individualmente na rede. As simulações são determinísticas por padrão, contudo há um conjunto de configurações que introduzem aleatoriedade no software.

As bibliotecas do SUMO são majoritariamente em C++. Algumas ferramentas de apoio foram codificadas em Python e XML.

No SUMO há a possibilidade de criar os próprios mapas através de códigos XML, importar mapas de outros softwares de simulação ou mesmo de sites como o Open-streetmaps. Isto é, cidades inteiras podem ter seus tráfegos simulados no SUMO.

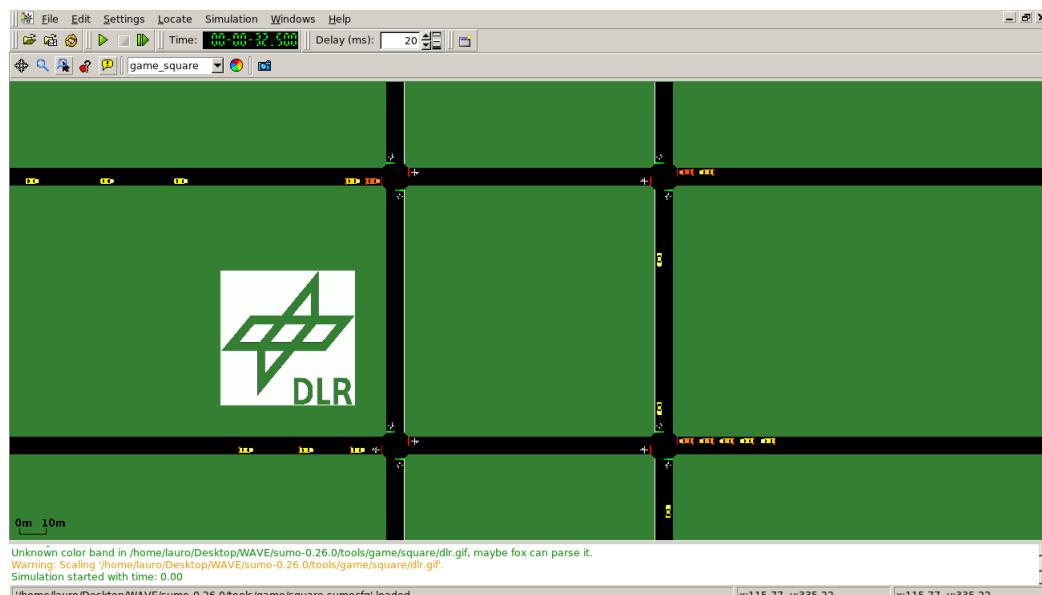


Figura 43 – Interface SUMO

Fonte:sumo.dlr.de

Além das características já citadas acima, o SUMO também permite:

- A criação de ruas de múltiplas faixas, com regras de trânsito específicas por faixa e com mudança de faixa permitida aos veículos;
- Regras e temporização de semáforos, sendo possível criar algoritmos que os controlem;
- Criação de fluxos e rotas para veículos seguirem;
- Escolha de diferentes tipos de veículos (carros *sedan*, *hatch*, ambulâncias, motos, ônibus, caminhões, etc)
- Configuração de regras de trânsito específicas (mão inglesa, vias com preferência, etc.)

Uma das características fundamentais do SUMO é que este pode interoperar com outras aplicações em tempo de execução transferindo dados da simulação para o processamento em outros sistemas. Isto também possibilita que o SUMO receba dados provenientes de outros sistemas, alterando diretamente no resultado de simulações e introduzindo mais um fator de aleatoriedade.

4.3 Middleware de Integração

Para integrar aplicações de simulações de rede e de tráfego com troca de informações bidirecional, é necessário ter um *framework* que faça a interface entre estes software e ofereça bibliotecas de suporte. O *framework* escolhido para esta função foi o VEINS, que será abordado com maiores detalhes na próxima seção.

4.3.1 Vehicles in Network Simulation

VEINS (*Vehicles in Network Simulation* ou Veículos em Simulação de Redes) é um *framework* para o OMNET++ que contém bibliotecas para simulação de redes veiculares e faz a comunicação entre o OMNET++ e o SUMO.

Este *framework* foi escolhido por oferecer as seguintes características:

- Possui código 100% aberto, promovendo extensibilidade irrestrita;
- Permite a reconfiguração de rotas dos veículos do SUMO em tempo de execução em reação ao envio de pacotes de rede;
- Baseado nos padrões IEEE 802.11p e IEEE 1609.4 para descrever as camadas física e de rede, incluindo operação multi-canal, QoS, efeitos de ruído e interferência;
- Inclui módulos de redes de celular, como LTE;

- Permite a importação de cenários do OpenStreetMap, incluindo prédios, limites de velocidade, semáforos, etc;
- Permite empregar módulos de atenuação de sinal causados por veículos e prédios;
- Gera dados com uma diversidade de métricas, incluindo tempo de trajeto e emissões de dióxido de carbono;
- Possui uma base ampla de usuários.

O VEINS utiliza uma relevante coleção de módulos aplicados em suas simulações de redes veiculares. Estes módulos servem como base para a construção de uma simulação específica, utilizando apenas aqueles relevantes para o cenário concebido. Alguns desses módulos são:

O MiXiM é utilizado para modelagem precisa da camada física, trabalhando com distribuição da potência de transmissão por tempo e distância;

O TraCI (*Traffic Control Interface* ou Interface de Controle de Tráfego) é uma interface que transforma os veículos do SUMO em nós de rede do OMNET++. Este módulo é utilizado pra monitorar o fluxo da simulação no SUMO, atualizando informações sobre a mobilidade do nó (posição, velocidade, direção) no OMNET++, baseado nos comportamentos dos veículos. Este módulo permite a reconfiguração de rotas dos veículos em tempo de execução e que VEINS e SUMO trabalhem em conjunto trocando mensagens de status através de uma porta TCP aberta;

Os módulos IEEE 802.11p e IEEE 1609.4 DSRC/WAVE realizam uma modelagem mais precisa das camadas física e de rede WAVE, incluindo técnicas de modulação e coordenação de canal, QoS conforme EDCA. Este módulo também inclui a manipulação de WSM, quadros de gerenciamento (*beacons*), WSAs, BSMs, etc.

Os módulos Two-Ray Interference gerencia a perda de pacotes em diferentes distâncias e trabalha com a propagação de mensagens entre os carros, a distância máxima de uma mensagem enviada por um único nó e a distância máxima da mesma mensagem propagada através de outros nós;

O módulo Obstacle Shadowing monitora os obstáculos que podem interferir a propagação da mensagem.

A vasta biblioteca do VEINS em C++ contém classes que exercem diversos papéis no contexto de simulações veiculares. O código fonte destas classes pode ser alterado ou

estendido para a obtenção de resultados mais específicos. Quando importados no OM-NET++, tais classes são encontradas no diretório src/veins/modules do projeto VEINS. Algumas classes relevantes deste *framework* são explicadas em detalhe na Tabela 7.

Classes	Detalhes
TraCICommandInterface	Classe que controla o comportamento dos veículos no SUMO. É possível enviar comandos de mudança de faixa, velocidade, rota, etc. Esta classe simula os comportamentos de um motorista.
TraCIScenarioManager	Classe que cria e gerencia o nó de rede de cada veículo.
TraCIMobility	Classe que controla a situação dos diversos sensores do veículo. É possível obter informações como quilometragem, velocidade, emissão de carbono, estado atual (em deslocamento, estacionando, parado, etc), entre outros. Esta classe simula o funcionamento dos diversos sensores de um carro.
Consts80211p	Arquivo cabeçalho que reúne constantes relacionadas aos padrões IEEE WAVE, como o tempo para alterar do modo transmissão (Tx) para recepção (Rx), intervalo de guarda, número dos canais, etc.
ConstsPhy	Arquivo cabeçalho que reúne constantes da camada física de um rádio. Contém informações sobre modulação de sinal, largura de banda, etc.
PhyLayer80211p	Classe que implementa módulos de modelos analógicos e faz pequenas alterações no módulo base MiXiM.
Mac1609_4	Classe que reúne as funções da camada de rede do modelo WAVE.
BaseWaveApplLayer	Classe que estabelece uma base para funcionamento de uma aplicação WAVE além de iniciar WSMs de forma similar à primitiva <i>WSM-WaveShortMessage.request</i> .
WaveShortMessage	Classe que reúne os campos do cabeçalho além dos dados gerados por uma aplicação WAVE.
ObstacleControl	Classe que cria obstáculos baseado no arquivo de polígonos *.poly.xml criado.

Tabela 7 – Módulos do VEINS

Para a realização das simulações veiculares, é necessário manipular arquivos específicos dentro da pasta da simulação. Os arquivos são listados em detalhe na Tabela 8.

Arquivos	Detalhes
omnetpp.ini	Arquivo de configuração de parâmetros. Valores diversos podem ser atribuídos para gerar resultados diferentes.
*.launchd.xml	Arquivo XML de configuração para ser enviado ao pequeno <i>daemon</i> dos módulos TraCI que realiza a execução em pares dos <i>frameworks</i> SUMO e OMNET++, rodando em plano de fundo.
*.ned	Arquivo de descrição de topologia da rede. Pode ser um nó, uma arquitetura de dispositivo, um protocolo, uma camada do modelo OSI, etc.
*.net.xml	Arquivo XML que contém o mapa utilizado como painel para a simulação.
*.poly.xml	Arquivo XML que contém mapa de polígonos. É relacionado ao arquivo de mapa *.net.xml. O mapa de polígonos é essencial na utilização do módulo de obstáculos do VEINS.
*.rou.xml	Arquivo XML que contém as rotas a serem utilizadas pelos veículos durante a simulação.
*.rou.alt.xml	Arquivo XML que contém rotas alternativas a serem utilizadas pelos veículos durante a simulação.
*.sumo.cfg	Arquivo XML que integra a simulação no SUMO. Arquivos *.net.xml, *.poly.xml, *.rou.xml, *.rou.alt.xml podem ser citados nele.
config.xml	Arquivo XML que contém módulos de modelos analógicos de sinal além da classe decisora (<i>decider</i>) para processamento de sinais utilizados pela simulação. Os modelos analógicos que podem ser utilizados estão contidos em src/veins/modules/analogueModel no projeto VEINS. Os modelos analógicos padrão são <i>SimplePathlossModel</i> e <i>TwoRayInterferenceModel</i> e em caso de simulação com obstáculos <i>SimpleObstacleShadowing</i> . O <i>decider</i> padrão é do tipo <i>Decider80211p</i> .

Tabela 8 – Arquivos do OMNET++

Portanto, conforme demonstrado neste capítulo, a escolha do OMNET + VEINS + SUMO neste trabalho de conclusão de curso se justifica, pois será possível criar aplicações para Redes Veiculares que operem de forma similar àquilo descrito na família de padrões 1609, simulando desde cenários com poucos veículos até situações mais complexas como grande tráfego em cidades conhecidas. Este conjunto de programas apresentam uma ampla variedade de logs, permitindo uma análise mais profunda da demonstração produzida, além de permitir diversas configurações de elementos reais como potência e atenuação do sinal, interferência por obstáculos, número de veículos e suas velocidades, produzindo, portanto, simulações bem próximas à realidade.

No próximo capítulo serão apresentados três cenários simulados nos softwares supracitados, utilizando-se de uma aplicação desenvolvida pelos autores deste trabalho, nos quais os resultados obtidos serão analisados e discutidos.

5 Especificações do Sistema e Estudo de Caso

Este capítulo apresenta um estudo de caso, tendo por objetivo experimentar através de simulações o uso de uma aplicação de emergência desenvolvida para comunicações interveiculares. A escolha do framework VEINS para a realização deste estudo de caso possibilitou ir além das simulações. Foi realizado neste TCC a inclusão de novas classes no código-fonte do *framework*, visando adaptar as simulações a realidade proposta neste trabalho.

Os experimentos mostrados a seguir têm todos os veículos equipados com tecnologias Car2X. Foi garantido que um mesmo conjunto de sementes aleatórias fosse gerado para os experimentos a serem comparados, o que descarta uma possível influência nos dados coletados e aumenta o grau de confiabilidade destes.

As simulações realizadas nos cenários subsequentes utilizam o mesmo espectro regulamentado pela FCC para comunicações DSRC nos Estados Unidos. A ANATEL (ATO No 2.099, 2012) ainda não estabeleceu um espectro para as comunicações interveiculares no Brasil até a presente data.

5.1 Especificações do Sistema

Conforme relatado na [seção 4.3](#), o *framework* escolhido para simulação de Redes Veiculares é o VEINS. Devido a grande quantidade de atributos e métodos presentes no código fonte deste *framework*, a [Figura 44](#) demonstra um diagrama de classes simplificado do funcionamento do software. A [Tabela 9](#) apresenta uma breve descrição sobre suas principais classes.

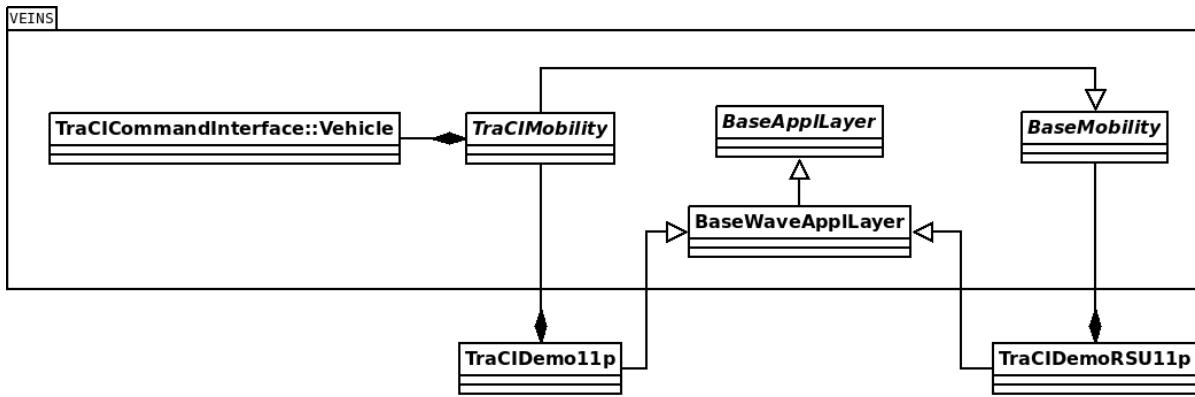


Figura 44 – Diagrama Simplificado de Classes do VEINS

Fonte: Autor

Vale ressaltar que este diagrama de classes é uma versão simplificada do diagrama original deste *framework*.

Classes	Descrição
BaseAppLayer	Base para uma camada de Aplicação Genérica. Implementa o envio de pacotes para a camada inferior.
BaseMobility	Base para todos os tipo de mobilidade para dispositivos Car2X. Esta base provê comportamento apenas para módulos estáticos como Unidades de Acostamento (RSUs).
TraCIMobility	Classe abstrata que gerencia mobilidade dinâmica para dispositivos Car2X móveis.
TraCICommandInterface::Vehicle	Implementação de mobilidade para Veículos. Permite alteração de velocidade, mudança de rota, e obtenção de dados sobre a posição atual.
BaseWaveAppLayer	Base para camada de Aplicação WAVE com funcionalidades específicas como a criação, envio e recebimento de WSMs.
TraCIDemo11p	Aplicação Exemplo para comunicação entre Veículos (OBUs).
TraCIDemoRSU11p	Aplicação Exemplo para Unidades de Acostamento (RSUs).

Tabela 9 – Descrição das Classes do VEINS

5.1.1 Aplicação de Emergência

Atualmente, os veículos de segurança pública utilizam-se apenas de giroscópio e sirene para solicitar passagem no trânsito, o que faz com que quaisquer barreiras visuais e sonoras dificultem a percepção de outros motoristas, possibilitando a análise de novos métodos de alerta que não sejam influenciados por estes aspectos.

As Redes Veiculares permitem um novo meio de comunicação no trânsito, proporcionado uma alternativa nos momentos em que o giroscópio e sirene não são suficientes. Por ser uma tecnologia que permite a comunicação em um ambiente de trocas rápidas de informação, os veículos comuns tomam ciência antecipa de fatos ocorridos no trânsito.

O código fonte do VEINS foi modificado, e uma aplicação de emergência foi desenvolvida para o estudo de caso neste trabalho. Esta aplicação objetiva ser de benefício para veículos de segurança pública, tais quais:

- Veículos de Socorro
- Veículos Policiais
- Veículos de Fiscalização e Operação de Trânsito
- Veículos de Agentes Públicos e Políticos

Esta aplicação, quando ativada, emite mensagens periódicas aos dispositivos Car2X nas redondezas. A partir do recebimento destas mensagens, os veículos comuns podem ser orientados para uma melhor tomada de decisão no trânsito.

Um diagrama de classes da aplicação de emergência foi desenhado e é apresentado na Figura 45.

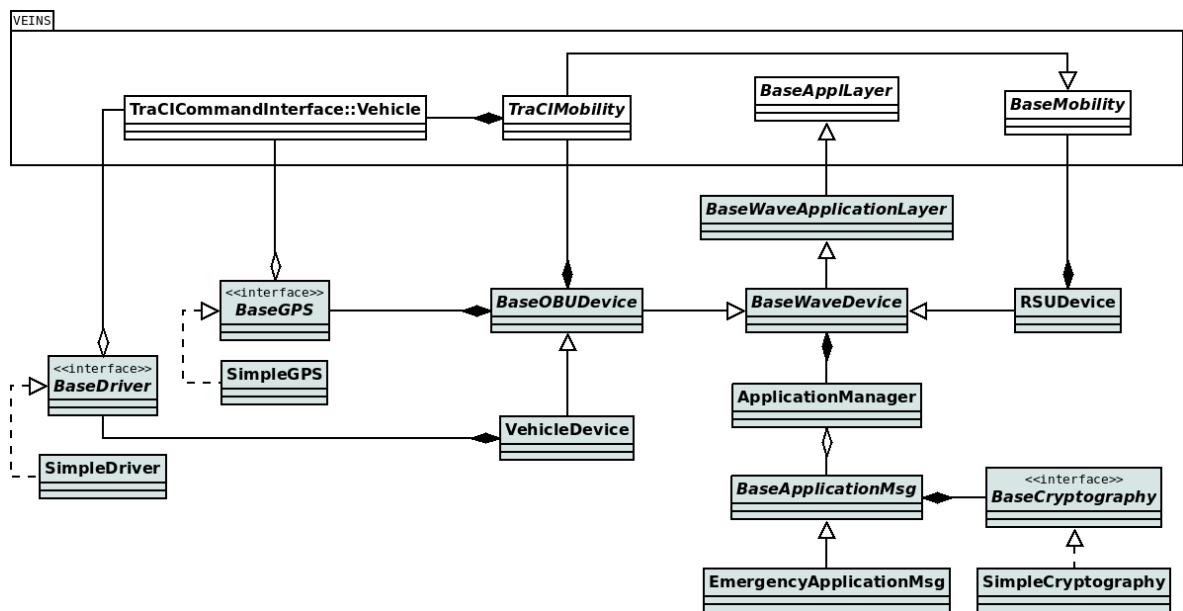


Figura 45 – Diagrama de Classes para o Estudo de Caso

Fonte: Autor

A Tabela 10 contém uma breve descrição das classes desenvolvidas para extensão do VEINS e para este estudo de caso.

Classes	Descrição
BaseWaveApplicationLayer, BaseWaveDevice	Classes inspiradas na classe <i>BaseWaveApplLayer</i> do VEINS. Esta classe foi refatorada com intuito de simplificar o entendimento e diminuir o acoplamento do código.
BaseDriver	Interface que implementa comportamentos de um motorista. A classe concreta <i>SimpleDriver</i> implementa um motorista simples com uma função simples de mudança de faixa. A base desta função foi adicionada em <i>TraCI-CommandInterface::Vehicle</i> , a fim de possibilitar a comunicação com o simulador de tráfego. Outros tipos de motoristas com comportamentos mais complexos podem ser implementados.
BaseGPS	Interface com funções específicas de GPS. A classe concreta <i>SimpleGPS</i> implementa um GPS simples com comportamentos básicos, como a obtenção e verificação de rotas de veículos.
BaseOBUDevice	Classe abstrata que contém comportamentos comuns em dispositivos Car2X móveis para veículos, como a inicialização de um GPS. A classe concreta <i>VehicleDevice</i> invoca um objeto motorista. Outros tipos de dispositivos Car2X móveis podem ser implementados para pedestres e ciclistas, por exemplo.
RSUDevice	Classe utilizada por Unidades de Acostamento. Instancia um objeto <i>BaseMobility</i> , característico de dispositivos estáticos.
ApplicationManager	Classe que gerencia aplicações inspirado no Padrão de Projeto Composite. O Gerente de Aplicações permite ao dispositivo WAVE executar diversas aplicações ao mesmo tempo, diferente dos exemplos do VEINS, onde cada dispositivo executa exclusivamente uma aplicação. <i>BaseApplicationMsg</i> é uma classe abstrata que contém informações comuns à aplicações WAVE. É possível a adição de novas classes no futuro como aplicações de clima, comboio de veículos, segurança no tráfego, etc.
BaseCryptography	Interface que contém a base para criptografia dos dados das aplicações. A classe concreta <i>SimpleCryptography</i> implementa uma criptografia simples como exemplo.
EmergencyApplicationMsg	Principal classe para o estudo de caso deste trabalho. Contém comportamentos de envio e recepção de mensagens de emergência. Requisita serviços de um GPS e contém o PSID 0x0C .

Tabela 10 – Descrição das Classes desenvolvidas

As mensagens da aplicação de emergência são enviadas no formato broadcast. Estas mensagens são de alcance direto (*single-hop* ou salto único) e geradas na camada de aplicação WAVE. Serão requisitados para o envio da mensagem apenas os serviços da camadas de aplicação, da subcamada de enlace MAC e da camada física. Os serviços de outras camadas, como a de transporte e a subcamada de enlace LLC são apropriados para o envio de outras naturezas de mensagens, como mensagens de protocolos roteamento que contém múltiplos saltos (*multihop*).

O tamanho dos dados nas mensagens de emergência é dinâmico. Para este estudo de caso, o campo *wsmData* contém o percurso parcial a ser percorrido pelo veículo de segurança pública. O formato deste campo é o seguinte:

$$(L_1)[R_1][R_2]\dots[R_n]$$

L_1 e R_1 representam respectivamente a faixa e rua em trânsito. R_2 até R_n simbolizam as próximas ruas a serem percorridas pelo veículo de segurança pública.

Devido à sua natureza, a aplicação de emergência utiliza o PSID **0x0C** no cabeçalho das mensagens, apropriado para situações de emergência. O código abaixo representa como essa informação é gerada pelo veículo de segurança pública através da função **prepareMessage**.

```
//numberOfRoads: Número máximo de ruas a ser obtido, inicializado no construtor
std::string data = gps->getCourse(numberOfRoads);
std::string encryptedData = cryptography->encrypt(data);

WaveShortMessage* wsm = new WaveShortMessage(*baseWsm);
wsm->setWsmData(encryptedData.c_str()());
wsm->setPsid(myPsid); //myPsid: inicializado no construtor com valor 0x0C
wsm->setChannelNumber(Channels::CCH);
wsm->addByteLength(encryptedData.size());

return wsm;
```

A aplicação de emergência solicita o percurso do veículo a um GPS (Rua Atual + próximas ruas) e encripta esta informação logo em seguida. Com o dado pronto para ser enviado, a aplicação inicia a preparação da mensagem tendo como base uma WAVE *Short Message* (WSM) inicializada pela classe *BaseWaveApplicationLayer*. Esta aplicação utiliza a classe *BaseWaveApplicationLayer* para incluir na WSM o dado encriptado, o PSID da aplicação e o número do canal a ser utilizado, que neste caso é o canal de controle.

5.2 Estudo de Caso

Para elucidação do tema proposto neste trabalho monográfico, foram criados neste estudo de caso 3 experimentos, que estão detalhados logo a seguir.

5.2.1 Descrição dos Experimentos

A aplicação de emergência descrita na seção anterior será utilizada e medida nos seguintes experimentos:

1. Análise de comunicações entre veículos em uma rodovia simples.
2. Análise de comunicações entre veículos e estruturas de acostamento em uma rodovia simples.
3. Análise de comunicações entre veículos e estruturas de acostamento na microrregião de Ipanema e Copacabana no Rio de Janeiro.

Os experimentos 1 e 2 contém simulações com cenários e métricas mais inteligíveis. Estes experimentos têm por objetivo elucidar o funcionamento das comunicações interveiculares, preparando o leitor para o experimento 3.

O conjunto de experimentos 3 apresenta medições mais complexas, como a análise de valores ótimos para parâmetros da aplicação de emergência utilizados por dispositivos Car2X em tráfego rodoviário na região dos bairros de Ipanema e Copacabana, no Rio de Janeiro.

Os experimentos citados serão frequentemente comparados com suas medições chamadas de “medições base”, as quais não executam a aplicação de emergência em seus cenários.

É necessário informar que os conjuntos de amostras em cada experimento compartilham das mesmas sementes aleatórias, o que aumenta a confiabilidade dos dados gerados.

5.2.2 Experimentos 1 e 2: Descrição do Cenário

O objetivo dos experimentos 1 e 2 é mostrar o funcionamento e a efetividade da aplicação desenvolvida em comunicações interveiculares.

Para a simular estes experimentos, um cenário comum foi desenvolvido através do software netedit, contido no pacote do *framework* SUMO. Foi desenhada uma rodovia de duas pistas com faixas duplas de mesmo sentido e 4000 metros de comprimento, conforme mostrado na [Figura 46](#).

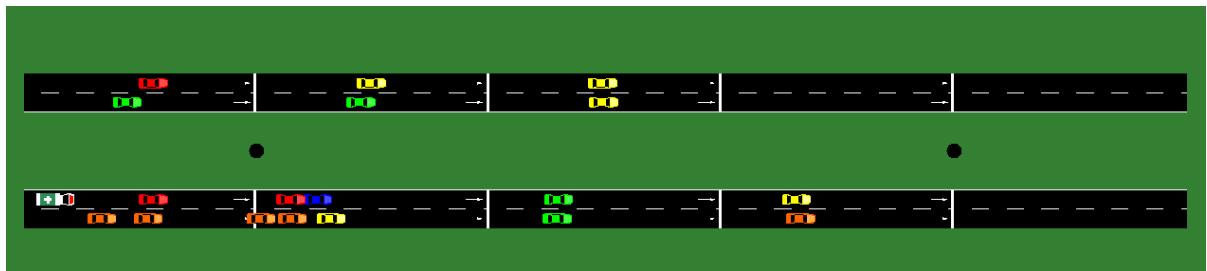


Figura 46 – Cenários dos Experimentos 1 e 2 (escala reduzida)

Fonte: Autor

Neste experimento, 19 veículos e 5 unidades de acostamento localizadas entre as pistas foram espalhadas pelo cenário. A Tabela 11 traz informações mais detalhadas sobre os dispositivos Car2X contidos neste cenário.

Dispositivos Car2X	Representado por	Potência de transmissão	Aplicações sendo executadas
secOBU[0]	Veículo de Resgate	20 mW	Aplicação de Emergência com módulo de envio periódico de mensagens.
OBU[0..18]	Veículo Vermelhos, Laranjas, Amarelos, Verdes e Azuis	20 mW	Aplicação Genérica de Emergência.
RSU[0..4]	Círculos Pretos	30 mW	Aplicação Genérica de Emergência autorizada a replicar mensagens.

Tabela 11 – Dispositivos Car2X no cenário dos experimentos 1 e 2

Nos experimentos 1 e 2, os valores dos parâmetros periodicidade do envio das mensagens e número de ruas foram definidos respectivamente como 3 segundos e 3 ruas.

Durante o experimento 1, as unidades de acostamento públicas permaneceram desligadas e apenas são avaliadas as mensagens trocadas entre veículos. No experimento 2, estas RSUs estão em funcionamento e replicam as mensagens recebidas pelo veículo de resgate. É válido ressaltar que a relação entre os valores de potência de transmissão e alcance do sinal são diretamente proporcionais. Portanto, as mensagens geradas por unidades de acostamento conseguem alcançar maiores distâncias se o valor da potência de transmissão for maior.

Conceitualmente, podemos dizer que os veículos comuns que receberem a mensagem de emergência poderiam representar as informações recebidas através de um mapa e emitir alertas sonoros em uma situação real. Estes alertas poderiam requisitar uma ação

do motorista para desobstruir a passagem do veículo de segurança pública caso este esteja nos arredores, conforme mostrado em conceito na [Figura 47](#).



Figura 47 – Computador de Bordo solicitando abertura de passagem (figura conceitual)

Fonte: Autor

5.2.2.1 Análise e Resultados: Experimento 1

O experimento 1 mede a influência da aplicação de emergência em comunicações de veículo para veículo no cenário proposto. Cada medição foi executada 100 vezes para reprodução de aleatoriedade e para posterior coleta e análise estatística.

Durante a simulação, pode-se perceber que as mensagens de broadcast do veículo de resgate alcançam também veículos comuns que estão em ruas paralelas, ou seja, fora de sua rota de colisão. Estes veículos recebem a mensagem porém não realizam qualquer ação, visto que estão em percursos diferentes em relação ao veículo de resgate. Isto é representado na [Figura 48](#) e na [Figura 49](#).

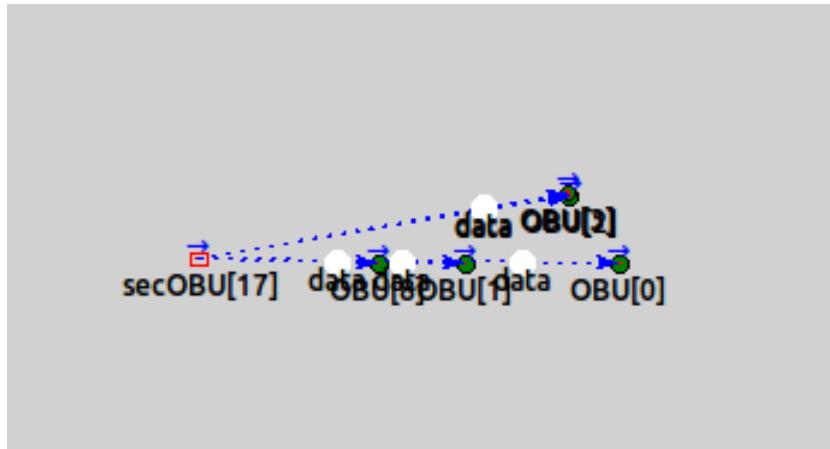


Figura 48 – Veículo de Resgate envia mensagem de broadcast

Fonte: TkEnv - OMNET++

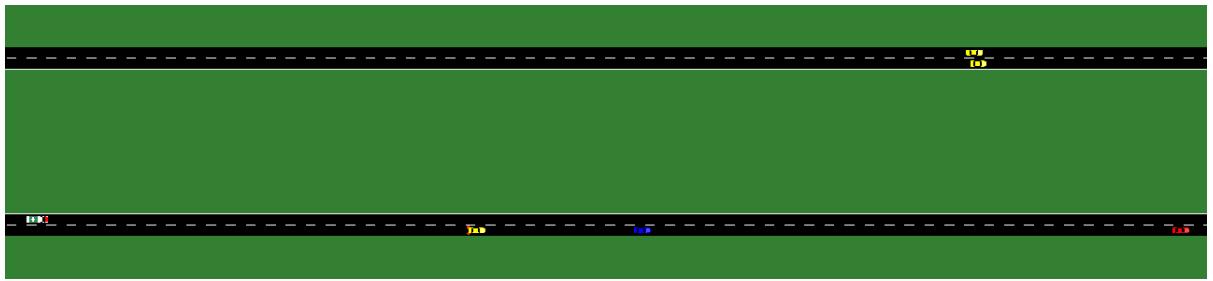


Figura 49 – Apenas os veículos da pista inferior mudam de faixa após mensagem

Fonte: SUMO-GUI

Os veículos que estão em rota de colisão com o veículo de resgate procuram o momento e espaço mais seguros para mudarem de faixa. Algumas ações são tomadas pelos motoristas destes veículos para evitar ao máximo o bloqueio da passagem. A Figura 50 mostra que no momento em que a mensagem de emergência alcança o veículo azul este está em menor velocidade em relação ao veículo vermelho. A mudança de faixa não pode ser realizada imediatamente após o recebimento da mensagem devido a uma possível colisão.



Figura 50 – Veículo azul em menor velocidade porém não é seguro mudar de faixa

Fonte: SUMO-GUI

Após alguns instantes, o veículo azul muda de faixa quando encontra uma distância segura para o veículo vermelho (Figura 51).

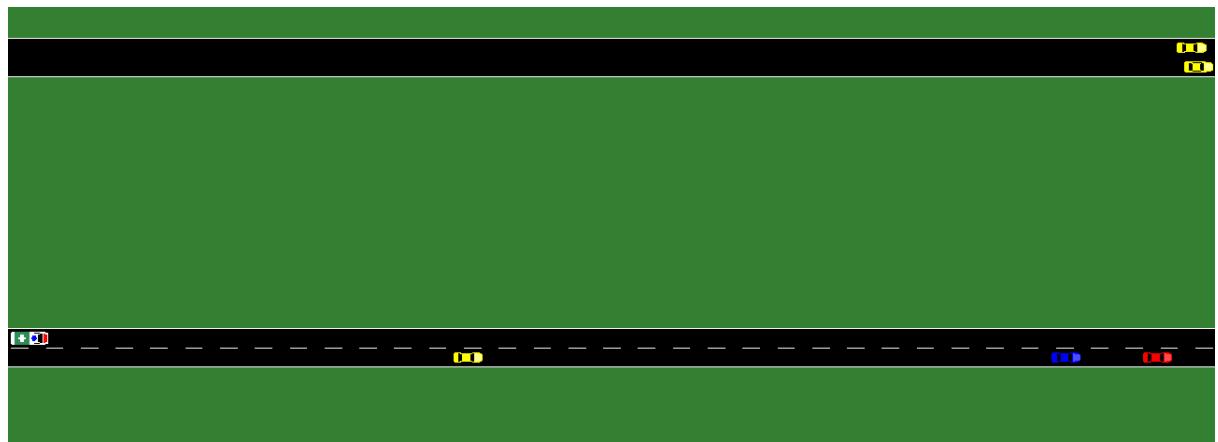


Figura 51 – Após alguns instantes a mudança de faixa é realizada

Fonte: SUMO-GUI

Outra situação observada no experimento 1 é vista na Figura 52. Embora em maior velocidade, o veículo vermelho é influenciado pela mensagem de emergência e não realiza a ultrapassagem a fim de não bloquear a rota do veículo de resgate.

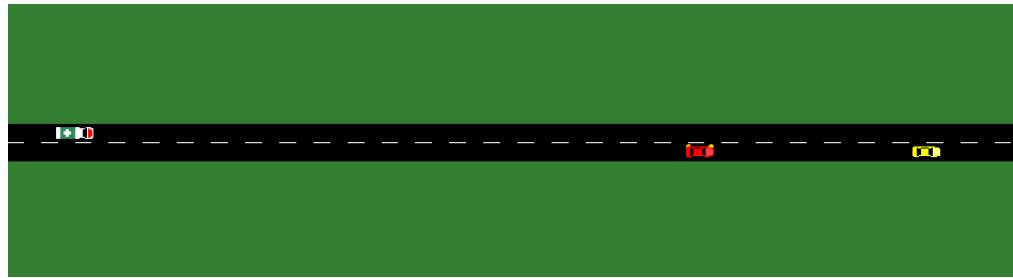


Figura 52 – Veículo vermelho é influenciado pela mensagem de emergência

Fonte: SUMO-GUI

O veículo vermelho, que após sofrer a ultrapassagem e não estar mais na rota de colisão do veículo de resgate, pode ultrapassar o veículo amarelo no momento em que encontra uma distância segura para a ação. A Figura 53 apresenta o momento deste acontecimento.



Figura 53 – Veículo vermelho muda de faixa e realiza a ultrapassagem

Fonte: SUMO-GUI

A Tabela 12 apresenta o tempo total de trajeto pelo veículo de resgate no experimento 1. O endereço para as tabelas que contém todos os valores da amostra está no Apêndice G.

Métricas	Medição base	Aplicação de emergência
Total percorrido	4000 m	4000 m
Tempo total gasto	128.62 s	120.82 s
Velocidade média	111,95 km/h	119,18 km/h

Tabela 12 – Resultados Experimento 1

É possível notar que devido a utilização da aplicação de emergência no cenário proposto, o veículo de resgate manteve uma velocidade média maior e pode reduzir o tempo total do trajeto em 6,07%.

5.2.2.2 Análise e Resultados: Experimento 2

O experimento 2 mede a influência da aplicação de emergência em comunicações entre veículos e unidades de acostamento no cenário proposto. Cada medição foi executada 100 vezes para reprodução de aleatoriedade e para posterior coleta e análise estatística.

Por ter uma potência de transmissão maior, as unidades de acostamento podem alcançar maiores distâncias do que as unidades de bordo. A figura [Figura 54](#) apresenta o raio máximo de alcance de cada RSU no cenário proposto. Cada RSU está no raio de alcance de outra unidade de bordo subsequente.

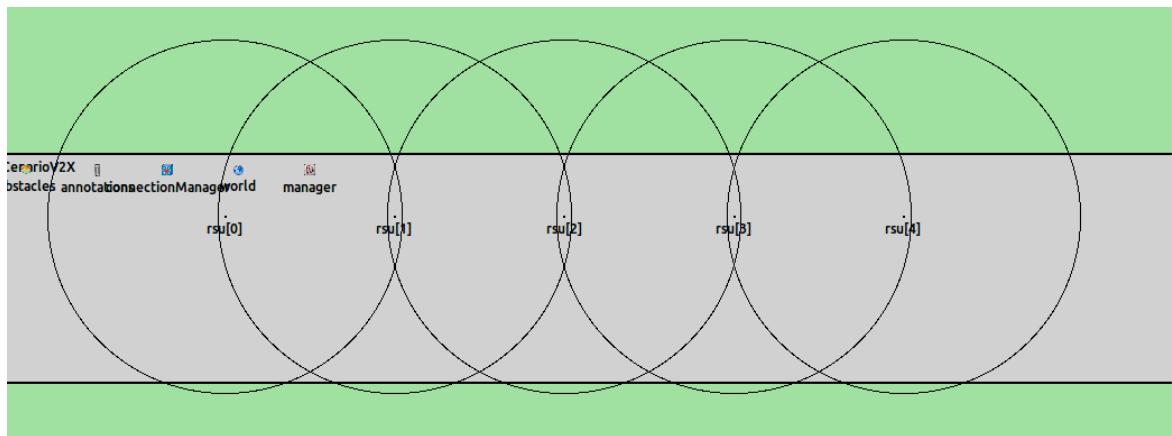


Figura 54 – Raio de alcance das RSUs

Fonte: TkEnv - OMNET++

A utilização de unidades de acostamento públicas é de relevante vantagem para a aplicação de emergência visto que sua mensagem alcança mais rapidamente um número maior de veículos comuns, permitindo a estes uma preparação mais antecipada à aproximação de algum veículo de segurança pública. Como exemplo, vemos na [Figura 55](#) que aproximadamente aos 14 segundos da simulação todos os veículos num raio de 4 quilômetros

ros já haviam recebido a mensagem da aplicação de emergência no experimento 2, enquanto apenas 5 veículos haviam recebido a mensagem de emergência no experimento 1.

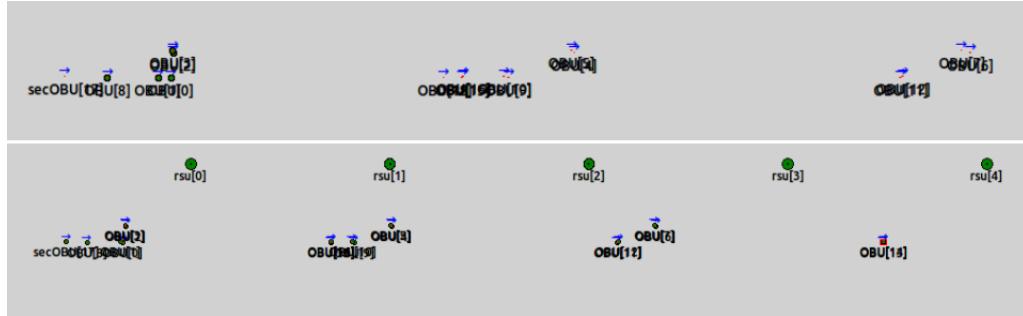


Figura 55 – Experimentos 1 e 2 - Comparaçāo do tempo de recebimento das mensagens

Fonte: TkEnv - OMNET++

A Tabela 13 apresenta o tempo total de trajeto pelo veículo de resgate no experimento 2. O endereço para as tabelas contendo todos os valores da amostra estão no Apêndice G.

Métricas	Medição base	Aplicação de emergência executada com RSUs
Total percorrido	4000 m	4000 m
Tempo total gasto	128.62 s	120,68 s
Velocidade média	111,95 km/h	119,2 km/h

Tabela 13 – Resultados Experimento 2

Nota-se que a utilização da aplicação de emergência e das unidades de acostamento públicas no cenário proposto fez o veículo de resgate manter uma velocidade média maior em relação ao experimentos 1, reduzindo o tempo total de trajeto do veículo de resgate em 6,17% em relação à medição base.

5.2.3 Experimento 3: Aplicação de Emergência na região de Ipanema-Copacabana

Para o terceiro cenário, um cenário mais complexo foi desenvolvido para a realização de experimentos. Foi importado o mapa do entorno da região de Ipanema-Copabana na cidade do Rio de Janeiro através do *OpenStreetMaps*. Esta região foi escolhida por serem dois bairros turísticos mundialmente famosos da cidade do Rio de Janeiro, além do fato deles também possuírem condições propícias à simulações de tráfego, como melhor planejamento urbano, número relevante de faixas por vias, semáforos, obstáculos, etc.

A Figura 56 apresenta o percurso do veículo de resgate, que parte da Avenida Vieira Souto em Ipanema e segue para Copacabana através da Avenida Nossa Senhora de Copacabana até seu ponto de destino. São percorridos aproximadamente 4800 metros neste trajeto, representado na imagem pela linha vermelha. Devido ao comportamento apresentado pelas unidades de acostamento no segundo experimento, 7 RSUs (pontos pretos Figura 56) foram espalhadas para auxiliar na propagação das mensagens da aplicação de emergência.



Figura 56 – Trajetória do veículo de resgate e RSUs

Fonte: SUMO-GUI

O volume de veículos gerado para as avenidas Vieira Souto e Nossa Senhora de Copacabana foi baseado em informações de tráfego obtidos em documento da Companhia de Engenharia de Tráfego do Rio de Janeiro (CET-Rio, 2014). A Tabela 14 apresenta os dispositivos WAVE contidos no experimento.

Dispositivos Car2X	Representado por	Potência de transmissão	Aplicações sendo executadas
secOBU[0]	Veículo de Resgate	20 mW	Aplicação de Emergência com módulo de envio periódico de mensagens.
OBU[0..114]	Veículo de diversas cores	20 mW	Aplicação Genérica de Emergência.
RSU[0..6]	Pontos Pretos	30 mW	Aplicação Genérica de Emergência autorizada a replicar mensagens.

Tabela 14 – Dispositivos Car2X no cenário dos experimento 3

Neste experimento foram incorporados modelos de atenuação de sinal por perda de caminho e reflexão (*TwoRayInterferenceModel*) e por obstáculos (*SimpleObstacleShadowing*) do framework VEINS para geração de dados mais próximos à realidade. Na figura Figura 57 são mostrados os raios de alcance de sinal das unidades de bordo contidas neste cenário. Cada RSU está no raio de alcance de outra unidade de acostamento subsequente.



Figura 57 – Disposição das RSUs no Experimento 3

Fonte: TkEnv - OMNET++

Foi estabelecida uma distribuição dos valores de velocidade máxima dos veículos comuns, sendo inferiores ao permitido em cada trecho. Apenas o veículo de segurança pública pode ultrapassar o limite máximo pois precisa percorrer o trajeto descrito no menor tempo possível.

A figura Figura 58 mostra uma parte do cenário Ipanema-Copacabana no SUMO-GUI. É necessário ressaltar que para este cenário todos os semáforos das ruas percorridas pelo veículo de resgate estão em funcionamento. O desenvolvimento de aplicações inter-

veiculares para controle e sincronismo de semáforos para liberação de vias a partir de mensagens de emergência pode ser discutido em trabalhos futuros.



Figura 58 – Fluxo dos Carros e obstáculos em vermelho no Experimento 3

Fonte: SUMO-GUI

Os dados coletados no experimento 3 tiveram seus *outliers* filtrados. Um *outlier* é descrito em estatística como um ponto que está muito distante das demais observações em uma série. Comumente é chamado de “ponto fora da curva”. A utilização da análise de *outliers* otimiza análises e garante confiabilidade na amostra.

O experimento tratado terá 95% de confiabilidade, isto é, serão analisados os 95% dos valores que encontram-se entre a média somada a duas vezes o desvio padrão e a média subtraída de duas vezes o desvio padrão.

5.2.3.1 Análise e Resultados: Experimento 3

Num conjunto de experimentos preliminar, foram medidos os parâmetros periodicidade e número de ruas da aplicação de emergência com o objetivo de encontrar seu valor ótimo para cenário Ipanema-Copacabana. Para o número de ruas, foi estabelecido um conjunto limitado de valores 2, 3, 4, 5. Já o parâmetro periodicidade detém um conjunto de valores 0.5, 1.0, 1.5, 2.0, 2.5 medidos em segundos.

Descobriu-se que os valores ótimos dos parâmetros para o cenário em questão são de 0,5 segundo de periodicidade e 4 ruas de abrangência. O endereço para as tabelas contendo todos os valores das amostras está no [Apêndice G](#). Vale ressaltar que estes valores não serão necessariamente ótimos para outros cenários. É necessário um estudo aprofundado para obtenção de valores ótimos que possam ser utilizados em quaisquer situações.

Os conjuntos de experimentos compartilham o mesmo conjunto de sementes aleatórias em cada medição, inclusive a medição base. Cada medição foi executada 100 vezes para

reprodução de aleatoriedade e para posterior coleta e análise estatística.

O comportamento da aplicação de emergência em unidades de bordo e de acostamento ocorre conforme discutido nos experimentos 1 e 2. As unidades de bordo de veículos comuns continuam apenas recebendo as mensagens de emergência enquanto as unidades de acostamento públicas podem receber e replicar as mensagens de emergência provenientes do veículo de resgate.

Na medição base, onde são mensurados os dados de cenários que não utilizam a aplicação de emergência, verificou-se que a média de tempo total de trajeto do veículo de resgate é de 471,53 segundos. O gráfico na [Figura 59](#) evidencia os valores medidos além de identificar os *outliers* nos círculos vermelhos.

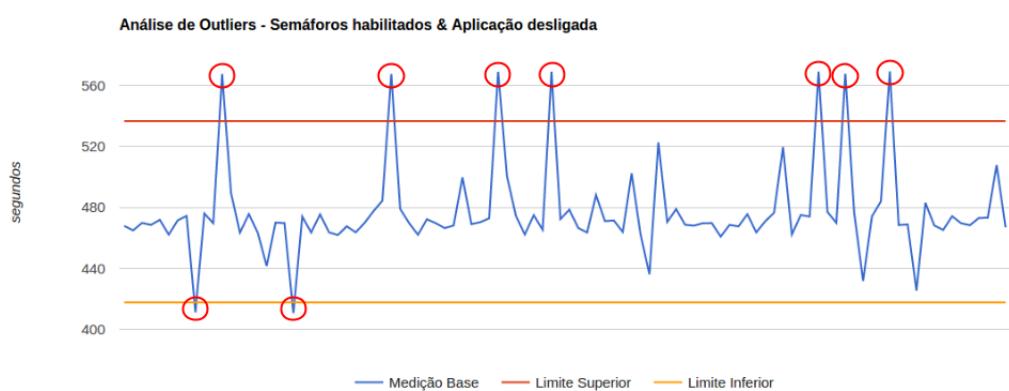


Figura 59 – Gráfico de Medições Base - *Outliers* nos círculos vermelhos

Fonte: Autor

Os valores coletados da medição base do cenário em questão estão dispostos na [Tabela 15](#).

Métricas	Medição base (semáforos habilitados)
Total percorrido	4800 m
Média de tempo total	477,11 s
Média após Análise de Outliers	471,53 s
Velocidade média	36,65 km/h

Tabela 15 – Resultados do Experimento 3 - Medições base

A segunda medição faz uso da aplicação de emergência. Com a utilização desta, verificou-se que a média de tempo total de trajeto do veículo de resgate é de 452,25 segundos. O gráfico contido na [Figura 60](#) evidencia os valores medidos além de identificar os *outliers*.



Figura 60 – Gráfico de Aplicação de Emergência ativada - *Outliers* nos círculos vermelhos

Fonte: Autor

Os valores coletados da medição com a utilização da aplicação de emergência estão dispostos na [Tabela 16](#).

Métricas	Medição base (semáforos habilitados)
Total percorrido	4800 m
Média de tempo total	458,06 s
Média após Análise de Outliers	452,25 s
Velocidade média	38,21 km/h

Tabela 16 – Resultados do Experimento 3 - Medição com aplicação ativada

Tais resultados apresentaram o veículo de resgate com uma rapidez de 4,09% quando a aplicação de emergência está em uso.

Durante a simulação, pode-se perceber uma “limitação” que não reproduz a realidade: O veículo de resgate respeitava completamente o tempo dos semáforos. Entretanto, o próprio código brasileiro de trânsito permite que os veículos de resgate avance o sinal vermelho do semáforo. Na prática, este veículo diminuiria sua velocidade e forçaria o avanço do sinal, desde que este ato pudesse ser realizado de forma segura.

Desta forma, a fim de executar uma análise mais crítica do experimento 3, consideramos um cenário ótimo onde os semáforos sempre estariam abertos na aproximação do veículo de resgate. Este cenário foi imaginado com o intuito de avaliar o tempo de resposta do veículo caso este não ficasse preso nos semáforos. A [Figura 61](#) apresenta o gráfico com os resultados obtidos.

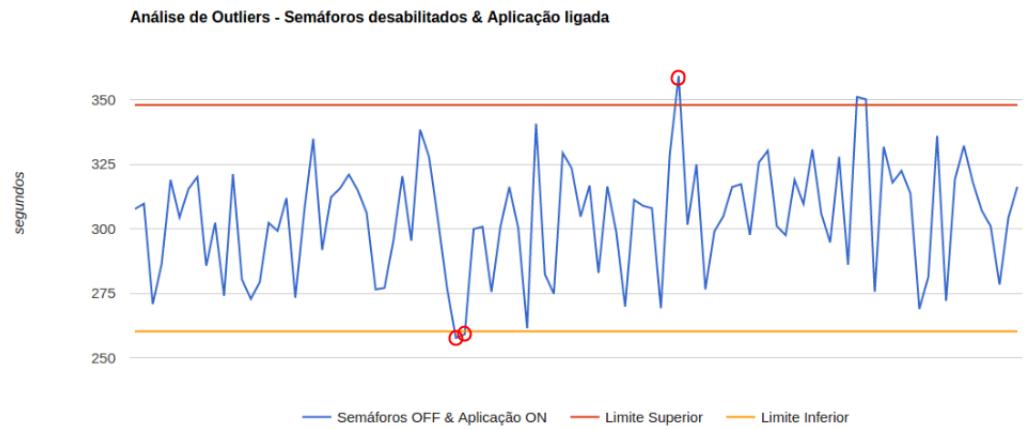


Figura 61 – Gráfico de Aplicação de Emergência ativada em cenário com semáforos abertos - *Outliers* nos círculos vermelhos

Fonte: Autor

Os valores coletados da medição com a utilização da aplicação de emergência e os semáforos abertos estão dispostos na [Tabela 17](#).

Métricas	Medição base (semáforos habilitados)
Total percorrido	4800 m
Média de tempo total	304,12 s
Média após Análise de Outliers	303,53 s
Velocidade média	56,93 km/h

Tabela 17 – Resultados - Medição com aplicação ativada e semáforos abertos

Analizando as medições com semáforos habilitados e desabilitados, podemos verificar que o uso da aplicação de emergência somado aos semáforos abertos resultou em uma rapidez de 35,63% no tempo total de percurso do veículo de resgate. Este cenário foi imaginado para avaliar uma situação mais próxima da realidade, onde os veículos de resgate avançam os sinais vermelhos. Em trabalhos futuros, é possível adicionar a comunicação entre os veículos de resgate e os semáforo (equipados com RSU) para que haja uma mudança no sincronismo dos sinais garantindo que não haja engarrafamento perto da aproximação do veículo de resgate.

5.3 Conclusão: Estudo de Caso

Podemos perceber uma melhora nos tempos totais de trajeto do veículo de resgate em todos os cenários. A performance da aplicação de emergência poderia trazer resultados

ainda mais significativos caso algumas limitações do software de simulação de tráfego fossem minimizadas.

No caso do software de simulação de tráfego SUMO, algumas destas limitações foram percebidas no curso deste estudo de caso, como por exemplo:

1. Não é permitida a ultrapassagem de semáforos fechados por quaisquer grupos de veículos. Esta situação, possível na vida real, é utilizada por alguns veículos de segurança pública caso estes estejam em serviço;
2. Não é atualmente implementada no SUMO a possibilidade de veículos percorrerem lado a lado em mesma faixa. Exemplo: Moto ultrapassa veículo comum sem mudar de faixa, percorrendo o trajeto ao lado deste veículo, mantendo uma distância segura;
3. Não é permitido atualmente no SUMO um veículo permanecer posicionado no meio de duas faixas. Esta manobra é muito usada pelos veículos de resgate, quando dois veículos que ocupam duas faixas vão para os extremos opostos de cada uma destas faixas para desobstrução de passagem;
4. No SUMO, os veículos não aceleram ou desaceleram antes de realizar a troca de faixa, ocasionando uma demora na ultrapassagem de veículos;
5. Não é permitido atualmente no SUMO um veículo realizar uma passagem pela contramão ou pela calçada, como é comumente visto em situações reais.

As limitações citadas influenciam a métrica de tempo total do veículo de resgate nos experimentos, ainda que a aplicação de emergência tenha ocasionado efeitos percebidos como melhor fluidez do trânsito gerando um tempo total menor. A [Tabela 18](#) apresenta os resultados dos experimentos do estudo de caso deste trabalho.

Veículo de Resgate	Tempo Total (Medição Base)	Tempo Total (Aplicação de Emergência)	rapidez em relação à medição base (%)
Experimento 1	128,62 s	120,82 s	6,07%
Experimento 2	128,62 s	120,68 s	6,17%
Experimento 3 (semáforos ligados)	471,53 s	452,25 s	4,09%
Experimento 3 (semáforos abertos)	471,53 s	303,53 s	35,63%

Tabela 18 – Resultados Experimentos do Estudo de Caso

Concluímos que a utilização de redes veiculares como um novo meio de comunicação no trânsito seria de relevante uso em eventuais desobstruções para veículos de segurança pública. Os resultados obtidos neste estudo de caso já demonstram que a utilização de mensagens de uma aplicação de emergência através de Redes Veiculares resultam em uma melhor fluidez no percurso dos veículos de resgate, de veículos policiais ou do corpo de bombeiros. Além do mais, cabe sempre ressaltar que qualquer ganho de tempo em um situação de resgate pode significar o ganho de uma vida.

As unidades de acostamento seriam de grande auxílio tanto para o bem público quanto na expansão do alcance de mensagens em Redes Veiculares.

Para a realização de simulações em redes veiculares, os softwares abertos de simulação de tráfego como o SUMO ainda necessitam de ajustes para atuar de forma fidedigna ao caráter real de sua atuação. Algumas melhorias que corrigem as limitações citadas já estão previstas pela equipe de desenvolvimento do SUMO¹.

No mais, as simulações de redes veiculares auxiliam relevantemente na preparação de provas de conceito, cooperando para uma abordagem analítica de cenários e para a economia de recursos.

¹ <http://sumo.dlr.de/trac.wsgi/roadmap> (acesso em 19 de set. 2016)

Conclusão

A padronização da tecnologia através das normas IEEE 1609 e ETSI mostra uma tentativa de incentivar a adoção de Rede Veiculares pelas indústrias. Tecnologias padronizadas facilitam a produção e implantação em maior escala.

A tecnologia Car2X não se apresenta apenas como futuro promissor e sim como uma realidade iminente. Nos últimos anos, inúmeras empresas de tecnologia voltaram sua atenção para a indústria automobilística a fim de desenvolver produtos com o objetivo de alcançar a tão desejada condução autônoma. CEOs de empresas automotivas repetem incessantemente que a inclusão de tecnologias eletrônicas e de comunicação sem fio em veículos é a tendência atual desta indústria².

No contexto das comunicações interveiculares, há um mercado promissor para empresas de desenvolvimento de software. Aplicativos para sensores, segurança, comunicação, entretenimento, informação, controle de tráfego poderão surgir, melhorando a experiência, segurança e a comodidade dentro do veículo.

No estudo de caso apresentado, vimos que uma aplicação de emergência alinhada à tecnologia Car2X facilita o trajeto de veículos de segurança pública. Estes veículos contam com uma nova forma de sinalização de sua presença, vencendo barreiras visuais e sonoras através da replicação de suas mensagens através da Rede Veicular.

² <http://telematicsnews.info/2016/06/20/toyota-connected-ceo-wants-to-contextualise-connected-cars-ju7204/> acesso em 23 de set. 2016

Trabalhos Futuros

O código desenvolvido para a aplicação de emergência apresentada no estudo de caso desta monografia permite ser incrementado em trabalhos futuros. Um mecanismo simples de criptografia foi utilizado como exemplo para realizar a encriptação dos dados da mensagem de emergência. Através da biblioteca crypto++, técnicas de criptografia podem ser incluídas para novos estudos de caso.

Atualmente, o código da aplicação contém orientação de mudança de faixa no corpo da mensagem de emergência. Novas orientações podem ser incluídas em trabalhos posteriores objetivando novos cenários. Como exemplo, mensagens contendo novas orientações como desaceleração, estacionamento, permissão de ultrapassagem, direção cautelosa, velocidades máxima e mínima podem ser criadas para cenários com outros fins. Novos estudos de caso poderiam simular a troca de mensagens em perseguições policiais, passagens de comitivas especiais, operações de blitz, sinalização vertical de vias, semáforos inteligentes, cruzamentos de tráfego intermodais (automóveis e trens, por exemplo), entre outros.

A classe Driver permite controlar ações típicas de um motorista humano no veículo. Esta poderia ser incrementada utilizando princípios de inteligência artificial. Como exemplo, as mudanças de faixa poderiam ser realizada ou não após o raciocínio de sinapses que calculam variáveis como distância entre veículos (frontal, traseira e lateral), aceleração e condições da pista.

Aplicações que comuniquem-se com semáforos possibilitariam melhor fluidez para os veículos de segurança pública caso estes pudessem alterar momentaneamente o sincronismo entre as fases do semáforo através da troca de mensagens de emergência.

Placas indicativas de velocidade máxima espalhadas em pontos estratégicos de uma rodovia poderiam ser equipadas de unidades de acostamento com o intuito de enviar WSMs periódicos indicando a velocidade adequada. Os dispositivos recebedores poderiam interpretar estas mensagens de diversas maneiras.

Radares poderiam ser equipados com RSUs para um registro mais confiável dos veículos que não estiverem em conformidade com as leis de trânsito, enviando a multa diretamente para o dono do veículo transgressor sem depender apenas de fotos, que por vezes não são suficientes.

Referências

- HARVEY, D. The Condition of Postmodernity: An Enquiry into the Origins of Cultural Change. Wiley-Blackwell, 1992. 392 p.
- WORLD HEALTH ORGANIZATION. Global Status Report on Road Safety, 2015. Relatório. Disponível em: <http://www.who.int/violence_injury_prevention/road_safety_status/2015/en/>. Acesso em: 2 de maio 2016.
- GAST, M. S. 802.11 Wireless Networks: The Definitive Guide. Segunda Edição. O'Reilly Media, 2005. 672 p.
- DUBUISSON, O. ASN.1 Communication between Heterogeneous Systems. Morgan Kaufmann, 2001. 562 p.
- FOROUZAN, B. A.; FEGAN, S. C. Data Communications and Networking. Quarta Edição. McGraw-Hill Higher Education, 2007. 1134p. (McGraw-Hill Forouzan Networking Series)
- DELGROSSI, L.; ZHANG, T. Vehicle Safety Communications: Protocols, Security, and Privacy. Nova Jérsia: Wiley, 2012. 372 p. (Information and Communication Technology Series)
- HARTENSTEIN, H.; LABERTEAUX, K. P. VANET: Vehicular Applications and Inter-Networking Technologies. Chichester: Wiley, 2010. 435 p. (Intelligent transportation systems)
- OLARIU, S.; WEIGLE, M. A. C. Vehicular Networks: From Theory to Practice. CRC Press, 2009. 472 p. (Chapman & Hall/CRC Computer and Information Science Series)
- BANKS, J. Discrete-Event System Simulation. Quarta Edição. Pearson Prentice Hall, 2005. 608 p. (Prentice-Hall international series in industrial and systems engineering)
- GIORDANO, S.; STOJMENOVIC, I. Position Based Routing Algorithms for Ad Hoc Networks: A Taxonomy. In: CHENG, X.; HUANG, X.; DU, D.-Z. (Eds.). Ad Hoc Wireless Networking. Kluwer Academic, 2003. p. 103-136.
- ROYER, E. M.; TOH, C.-K. A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks. IEEE Personal Communications, v. 6, n. 2, p. 46-55, 1999.

- CHENG, J., et al. Routing in Internet of Vehicles: A Review. *IEEE Transactions on Intelligent Transportation Systems*, v. 16, n. 5, p. 2339-2352, 2015.
- SOMMER, C.; GERMAN, R.; DRESSLER, F. Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis. *IEEE Transactions on Mobile Computing*, v. 10, n. 1, p. 3-15, 2011.
- CHENG, L., et al. Multi-Path Propagation Measurements for Vehicular Networks at 5.9 GHz. In: *WIRELESS COMMUNICATIONS AND NETWORKING CONFERENCE*, 2008, Las Vegas.
- YANG, F., et al. An Overview of Internet of Vehicles. *China Communications*, v.11, n. 10, p. 1-15, 2014.
- PEGDEN, C. D.; SADOWSKI, R. P.; SHANNON, R. E. *Introduction to Simulation Using Siman*. Segunda Edição. McGraw-Hill, 1995. 660 p. (Industrial engineering series)
- PURSULA, M. Simulation of Traffic Systems: An Overview. Disponível em: <http://publish.uwo.ca/~jmalczew/gida_5/Pursula/Pursula.html>. Acesso em: 6 de jun. 2016.
- KRAJZEWICZ, D., et al. Recent Development and Applications of SUMO: Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, v. 5, n. 3-4, p. 128-138, 2012.
- JAKOB, E. Lane-Changing Model in SUMO. *Proceedings of the SUMO2014 Modeling Mobility with Open Data*, Berlim, v. 24, p. 77-88, 2014.
- OMNeT++ Discrete Event Simulator: What is OMNeT++. Disponível em: <<https://omnetpp.org/intro/what-is-omnet>>. Acesso em: 6 de jun. 2016.
- Documentation: Veins. Disponível em: <<http://veins.car2x.org/documentation/>>. Acesso em: 6 de jun. 2016.
- GDAL: GDAL - Geospatial Data Abstraction Library. Disponível em: <<http://www.gdal.org/>>. Acesso em: 7 de jun. 2016.
- OSGeo.org: Your Open Source Compass. Disponível em: <<http://www.osgeo.org/>>. Acesso em: 7 de jun. 2016.
- Fox-toolkit.org. Disponível em: <<http://www.fox-toolkit.org/>>. Acesso em: 7 de jun. 2016.
- Proj.4: proj.4 4.9.3 documentation. Disponível em: <<http://proj4.org/>>. Acesso em: 7 de jun. 2016.

Xerces-C++ XML Parser. Disponível em: <<http://xerces.apache.org/xerces-c/>>. Acesso em: 7 de jun. 2016.

VARGA, A. OMNeT++ Installation Guide: Version 5.0. Disponível em: <<https://omnetpp.org/doc/omnetpp/InstallGuide.pdf>>. Acesso em: 7 de jun. 2016.

Tutorial: Veins. Disponível em: <<http://veins.car2x.org/tutorial/>>. Acesso em: 7 de jun. 2016.

Installing: Sumo. Disponível em: <<http://sumo.dlr.de/wiki/Installing>>. Acesso em: 7 de jun. 2016.

ANATEL. Plano de Atribuição, Destinação e Distribuição de Faixas de Frequências no Brasil: Ato n. 2099. Brasil, 2012. 175 p.

IEEE COMPUTER SOCIETY. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Part 11, 802.11-2012. Nova York, 2012. 2793 p.

IEEE COMPUTER SOCIETY. IEEE Guide for Wireless Access in Vehicular Environments (WAVE): Architecture, 1609.0-2013. Nova York, 2013. 78 p.

IEEE COMPUTER SOCIETY. IEEE Standard for Wireless Access in Vehicular Environments: Security Services for Applications and Management Messages, 1609.2-2016. Nova York, 2016. 240 p.

IEEE COMPUTER SOCIETY. IEEE Standard for Wireless Access in Vehicular Environments (WAVE): Networking Services, 1609.3-2016. Nova York, 2016. 160 p.

IEEE COMPUTER SOCIETY. IEEE Standard for Wireless Access in Vehicular Environments (WAVE): Multi-Channel Operation, 1609.4-2016. Nova York, 2016. 94 p.

IEEE COMPUTER SOCIETY. IEEE Standard for Wireless Access in Vehicular Environments (WAVE): Over-the-Air Electronic Payment Data Exchange Protocol for Intelligent Transportation Systems (ITS), 1609.11-2010. Nova York, 2010. 62 p.

IEEE COMPUTER SOCIETY. IEEE Standard for Wireless Access in Vehicular Environments (WAVE): Identifier Allocations, 1609.12-2016. Nova York, 2016. 21 p.

IEEE COMPUTER SOCIETY. Standard for Wireless Access in Vehicular Environments: Security Services for Applications and Management Messages Amendment - Certificate Management, P1609.2a. Nova York, 2010. 62 p.

IEEE SA - P1609.2a - Standard for Wireless Access in Vehicular Environments–Security Services for Applications and Management Messages Amendment - Certificate Management. Disponível em:
[<https://standards.ieee.org/develop/project/1609.2a.html>](https://standards.ieee.org/develop/project/1609.2a.html). Acesso em: 21 de jun. 2016.

IEEE SA - P1609.6 - Standard for Wireless Access in Vehicular Environments (WAVE) - Remote Management Service. Disponível em:
[<https://standards.ieee.org/develop/project/1609.6.html>](https://standards.ieee.org/develop/project/1609.6.html). Acesso em: 21 de jun. 2016.

SAE INTERNATIONAL. DSRC Implementation Guide: A guide to users of SAE J2735 message sets over DSRC. 2010. 210 p.

Features: Veins. Disponível em: [<http://veins.car2x.org/features/>](http://veins.car2x.org/features/). Acesso em: 19 de jul. 2016.

DIRETORIA DE DESENVOLVIMENTO GERÊNCIA DE INFORMAÇÕES DE TRÁFEGO. Volume Diário De Veículos Das Principais Vias Do Município Do Rio De Janeiro. Disponível em: [<http://www.rio.rj.gov.br/dlstatic/10112/5112752/4131653/VolumedasprinciaisviasdoRiodeJaneiro.pdf>](http://www.rio.rj.gov.br/dlstatic/10112/5112752/4131653/VolumedasprinciaisviasdoRiodeJaneiro.pdf). Acesso em: 25 de ago. 2016.

WHYTE, W., et al. Threat and Countermeasures Analysis for WAVE *Service Advertisement*. IEEE 18th International Conference on Intelligent Transportation Systems, Las Palmas, p. 1061-1068, 2015.

Anexos

ANEXO A – *Abstract Syntax Notation One*

Em telecomunicações, ASN.1 (*Abstract Syntax Notation One* ou Notação para Sintaxe Abstrata 1) é uma notação de comunicação padronizada internacionalmente em 1984 por um conjunto de organizações e funciona independentemente de hardware, sistemas operacionais e linguagens de programação utilizados. Esta notação descreve estruturas de dados usadas para a representação, codificação, transmissão e decodificação dos dados em redes de telecomunicações. Por estes motivos, a notação ASN.1 permite que a estrutura da informação entre sistemas diversos (torres de celulares, radiotelescópios, carros, *smartphones*, computadores, etc) seja compartilhada sem necessidade de saber como a mensagem é construída ou que meio é utilizado para a propagação (ar ou cabo).

ASN.1 define uma sintaxe abstrata da informação mas não restringe como a informação é codificada. Por isso, diversas regras de codificação como BER, CER e PER utilizam da sintaxe abstrata descrita em ASN.1. Regras de codificação descrevem como os valores definidos na notação podem ser codificados para a transmissão. Em outras palavras as regras de codificação ditam como a informação será traduzida em bytes para comunicações com ou sem fio. Diversos protocolos de aplicação como LDAP, H.323 e Kerberos também utilizam a notação ASN.1 em unidades de protocolos de dados trocados.

A ASN.1 envia informação compressa em qualquer forma (áudio, vídeo, dados, etc) para qualquer lugar que a comunicação digital é alcançada. Esta notação dispõe de uma tipagem básica pré-definida - os tipos inteiro, booleano e string existem na notação - e também permite a criação de novos tipos de dados. Esta característica é de extrema importância para a comunicação entre sistemas novos e antigos. Apesar de demonstrar semelhanças, a notação ASN.1 não é considerada uma linguagem de programação pois não possui operadores.

As informações na notação ASN.1 podem ser facilmente mapeadas em uma estrutura de dados de outras linguagens de programação, como Java ou C++. Assim, a informação da notação ASN.1 pode ser utilizada pelo código de uma aplicação para diversas finalidades.

ANEXO B – Management Information Base

A Base de Gerenciamento de Informação (MIB) é responsável por agrupar parâmetros operacionais e de configuração de um dispositivo de rede em objetos. A MIB cria uma coleção de objetos nomeados, seus tipos, e seus relacionamentos hierárquicos em uma entidade gerente ou em um protocolo. Como exemplo, a MIB do protocolo de transporte UDP (*User Datagram Protocol*) poderia ter o parâmetro *udpInDatagram* do tipo inteiro para indicar o número de datagramas UDP de entrada para o dispositivo.

A MIB é especificada utilizando notação ASN.1 e assemelha-se a uma base de dados. Seus identificadores de objetos (*Object Identifiers* ou OIDs) imitam uma árvore na sua estrutura hierárquica. A Figura 62 representa uma MIB com suas OIDs numa exibição de árvore.

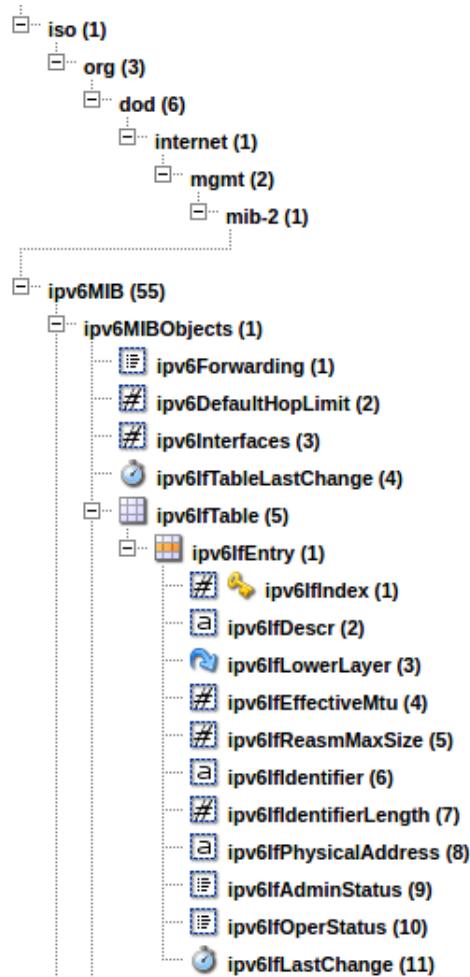


Figura 62 – MIB IPv6

Fonte: <http://support.ipmonitor.com/> acesso em 29 de jul. 2016

As camadas físicas e de enlace, por exemplo, têm sua própria MIB. Há entidades conceituais que gerenciam informações das camadas, requisitando informações e solicitando alterações na MIBs de outras camadas através de pontos de acesso a serviços (SAPs ou *Service Acess Points*), que funcionam como interfaces. Como exemplo, algumas alterações na camada MAC podem necessitar de alterações na camada física, portanto o SAP que fica entre a subcamada MAC e a camada física permite à entidade da subcamada MAC solicitar à entidade da camada física alterações na MIB. A [Figura 63](#) mostra um modelo conceitual contendo SAPs entre entidades e camadas.

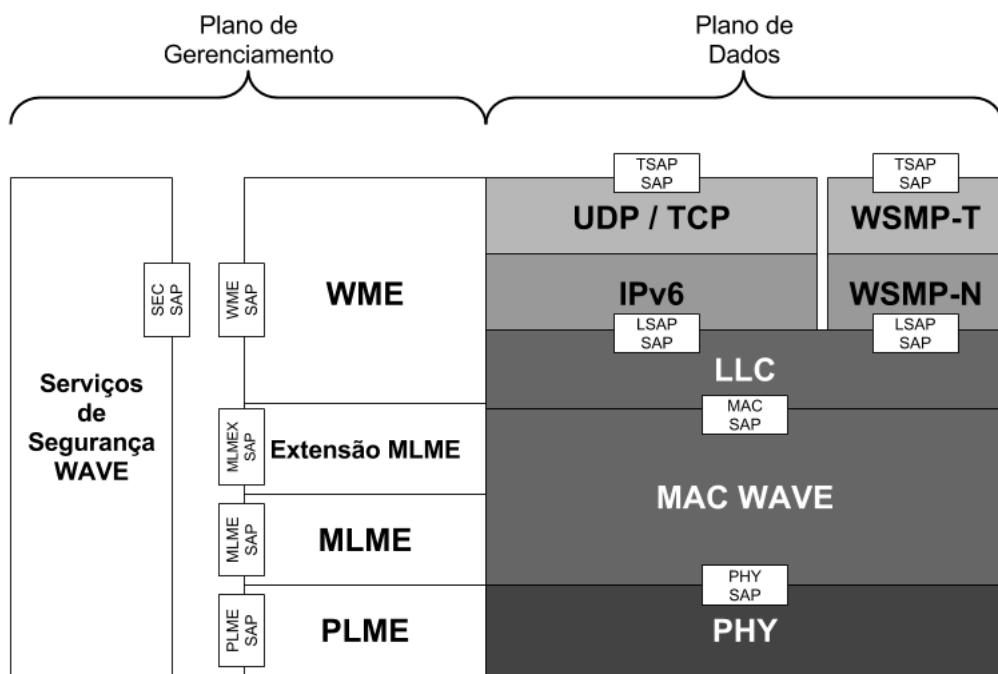


Figura 63 – MIBs - Interfaces entre camadas

Fonte: Autor

A MIB é um dos componentes essenciais no gerenciamento de redes, juntamente com a Estrutura de Gerenciamento de Informação (SMI) e o protocolo SNMP.

O próximo capítulo irá abordar brevemente a história das Redes Veiculares, apresentar a família de padrões WAVE e explicar em detalhes a pilha de protocolos WAVE.

ANEXO C – Alocação de PSIDs

Na família de padrões WAVE são alocados identificadores a fim de identificar dispositivos, serviços, organizações e tipos de mensagens WAVE segundo seus propósitos. A norma que aloca os identificadores de dispositivos ou serviços WAVE é a IEEE 1609.12

Para provedores de serviço, a identificação (*Provider Service Identifier* ou PSID) é um número inteiro de 4 octetos que varia entre 0 a 270 549 119 (em hexadecimal 10-20-40-7F). O PSID é um identificador de uma área de aplicação e tem 3 principais usos:

- Conforme o padrão WAVE de Serviços de Rede (IEEE 1609.3), um provedor de serviço identifica os anúncios de serviço “divulgados” pelos valores PSID contidos nas mensagens WSM transmitidas.
- Conforme o padrão WAVE de Serviços de Rede (IEEE 1609.3), o protocolo WSMP entrega conteúdo de mensagens WSM para entidades de camadas superiores baseado no valor PSID estabelecido pelo emissário no cabeçalho da mensagem.
- Conforme o padrão WAVE para Serviços de Segurança (IEEE 1609.2), um certificado de segurança lista o(s) valor(es) PSID identificando áreas de aplicação onde um emissário é autorizado a gerar unidades de dados de protocolo seguros e digitalmente assinados.

A atribuição de identificadores PSID para Redes Veiculares é coordenada por Entidades Verificadoras e é compartilhada com a identificação de aplicações para Sistemas de Transporte Inteligentes, descrita na norma ISO TS 17419. O formulário de registro de PSID é preenchido no site <https://standards.ieee.org/develop/regauth/psid>.

Na [Tabela 19](#) seguem exemplos de conjuntos de PSIDs alocados para diversas áreas de aplicação. A codificação dos bits do primeiro octeto e o estabelecimento de significância não são abordados neste trabalho.

PSID	p-PSID	Área de aplicação	Organizações ou Documentação
0x00	0p00	Sistema	ISO 15628
0x01	0p01	Pagamento Eletrônico de Taxas	ISO 14906
0x03	0p03	Transporte Público	ISO 15628
0x04	0p04	Informações de Tráfego para Viajantes	ISO 15628
0x05	0p05	Controle de Tráfego	ISO 15628
0x06	0p06	Gestão de Estacionamento	ISO 15628
0x07	0p07	Banco de Dados Geográfico de Estradas	ISO 15628
0x0C	0p0C	Aviso de Emergência	ISO 15628
0x20	0p20	Segurança e Conscientização V2V	SAE J2735
0x22	0p22	Segurança e Conscientização em veículos sobre trilhos (VLTs ou trens)	SAE J2735
0x23	0p23	Gestão de Segurança WAVE	IEEE 1609.2
0x7F	0p7F	Testes	IEEE 1609
0x82	0p80-02	Segurança e Conscientização em Interseções	SAE J2735
0x83	0p80-03	Sinalização Rodoviária e Informações ao viajante	SAE J2735
0x84	0p80-04	Trocas de dados de sensores veiculares	SAE J2735
0x87	0p80-07	WAVE Service Advertisement	IEEE 1609.3
0x10-20-40-7E	0pEF-FF-FF-FE	Roteamento IPv6	IEEE 1609.3

Tabela 19 – PSID - Alocação de Identificadores

ANEXO D – Informações do Elemento WAVE

Partes do cabeçalho WSMP e WSA permitem a presença do campo Informações Extensíveis do Elemento WAVE que deverá conter a estrutura apresentada na [Figura 64](#).

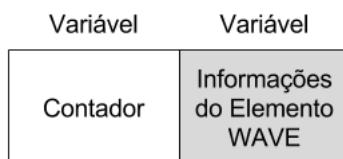


Figura 64 – Estrutura do campo Informações Extensíveis do Elemento WAVE

Fonte: Autor

O campo Contador informa quantas informações de Elementos WAVE existem dentro dessa extensão. No campo Informações do Elemento WAVE, existirão vários (valor do Contador) elementos respeitando a estrutura da [Figura 65](#).

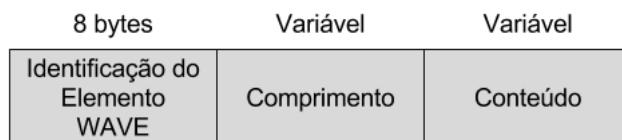


Figura 65 – Estrutura de uma Informações do Elemento WAVE

Fonte: Autor

O campo Identificação do Elemento WAVE revela o propósito do elemento. Como exemplo, o valor 9 neste campo indica que o conteúdo é um IPv6, enquanto o valor 5 indica uma localização 2D. A descrição de cada valor e o cabeçalho ao qual pertencem estão contidos na [Tabela 20](#). O campo Comprimento indica o tamanho do Informações do Elemento WAVE. O campo Conteúdo envolve o conteúdo ou valor da Identificação do Elemento WAVE em questão.

ID	Nome	Presente em
0 - 3	—	Reservado
4	Potência de Transmissão Usada	Cabeçalho WSMP-N
5	Localização 2D	Cabeçalho WSA
6	Localização 3D	Cabeçalho WSA
7	Identificador do Anunciante	Cabeçalho WSA
8	Contexto do Provedor de Serviço	Segmento de Informações do Serviço WSA
9	Endereço IPv6	Segmento de Informações do Serviço WSA
10	Porta do Serviço	Segmento de Informações do Serviço WSA
11	Endereço MAC do Provedor	Segmento de Informações do Serviço WSA
12	Parâmetros EDCA	Segmento de Informações do Canal WSA
13	DNS Secundário	WRA WSA
14	Endereço MAC do Gateway	WRA WSA
15	Número do Canal	Cabeçalho WSMP-N
16	Taxa de Transferência	Cabeçalho WSMP-N
17	Taxa de Repetição	Cabeçalho WSA
18	—	Reservado
19	Limite da Potência do Canal Especificado	Segmento de Informações do Serviço WSA
20	Limite do Contador WSA	Segmento de Informações do Serviço WSA
21	Tipo de Acesso ao Canal	Segmento de Informações do Canal WSA
22	Limite do Contador de Intervalo WSA	Segmento de Informações do Serviço WSA
23	Carga do Canal	Cabeçalho WSMP-N
24 - 255	—	Reservado

Tabela 20 – Descrição dos diversos Identificadores do Elemento WAVE

ANEXO E – Preparando o ambiente para simulações veiculares no VEINS

Um dos objetivos deste trabalho monográfico é mostrar como simulações em redes veiculares são realizadas e como desenvolver novas simulações para futuros estudos de caso.

No [Capítulo 4](#) foram definidos os *frameworks* a serem utilizados para a criação de estudo de caso baseado-se num conjunto de critérios.

O *framework* OMNET++ é o responsável pela simulação de eventos discretos, contendo bibliotecas para a simulação de redes de comunicação.

O *framework* SUMO gerencia o tráfego em modelo microscópico, emitindo mensagens de status de cada veículo em determinado intervalo de tempo.

O *framework* integrador VEINS recebe as mensagens de status do SUMO, gerando traços dos veículos no ambiente gráfico. O VEINS também envia mensagens através da interface de comunicação TRACI. Estas mensagens podem ou não alterar a rotas dos veículos em simulação no SUMO.

Para a correta instalação e síncrona comunicação dos *frameworks*, é necessária a instalação seguindo a ordem proposta além de componentes extras para que os *frameworks* tenham um funcionamento adequado.

Foram elaborados dois tutoriais para dois sistemas operacionais distintos: Windows 10 e Ubuntu 16.04. Encorajamos a instalação dos *frameworks* no sistema operacional Ubuntu para melhor *performance* da instalação e das simulações.

O download do SUMO pode ser feito através do site http://sumo.dlr.de/wiki/Main_Page, escolhendo a versão compatível com o sistema operacional desejado.

O download do OMNET++ pode ser feito através do site <https://omnetpp.org/> [omnetpp](#), escolhendo a versão compatível com o sistema operacional desejado.

O download do VEINS pode ser feito através do site <http://veins.car2x.org/> [download/](#).

Recomenda-se manter os arquivos descompactados em uma pasta padrão. Neste tutorial vamos assumir que esta pasta é nomeada **WAVE**. Determine qual sistema operacional será utilizado e siga os passos conformemente.

E.1 Sistemas Linux

1º Passo

Após os arquivos compressos do SUMO, OMNET++ e VEINS terem sido obtidos em seus respectivos sites, organize os dentro da pasta padrão **WAVE**.

Dependências

Para a correta simulação de redes veiculares, o *framework* SUMO deve possuir um conjunto de bibliotecas para realizar operações de maneira apropriada. Estas bibliotecas estão contidas na [Tabela 21](#).

Bibliotecas	Descrição
GCC	GCC (<i>GNU Compiler Collection</i>) é uma coleção de compiladores para Unix. Contém compiladores para linguagens como C, C++ e Java, além de diversas outras.
GDAL	<i>Geospatial Data Abstraction Library</i> (Biblioteca Abstrata de dados Geoespaciais) é uma biblioteca para tradução de formatos de dados geoespaciais vetoriais e matriciais. Tem seu código fonte aberto disponibilizada pela <i>Open Source Geospacial Foundation</i> (Fundação Geoespacial de Código Aberto). Esta biblioteca oferece um modelo de dados abstrato para matrizes e vetores a fim de fornecer estes dados nos mais diversos diferentes formatos à aplicações que os invocam.
FOX	Biblioteca baseada em C++ e voltada para o desenvolvimento de interfaces gráficas para o usuário.
PROJ.4	API (<i>Application Program Interface</i>) em C para realizar conversões entre projeções cartográficas. Esta biblioteca converte coordenadas geográficas de latitude e longitude em coordenadas cartesianas e vice-versa.
Xerces-C++	Manipulador XML escrito em C++. Através de sua biblioteca, códigos XML são gerados, manipulados e validados.

Tabela 21 – Bibliotecas necessárias para Sistemas Unix

2º Passo

Em muitos distribuições *Unix*, as bibliotecas e APIs mencionadas na [Tabela 21](#) já são partes do sistema e não necessitam ser instaladas. Recomenda-se, portanto, verificar se tais pacotes são padrão em sua distribuição.

3º Passo

É necessário instalar um pacote que gera *scripts* de configuração automaticamente para softwares em sistemas operacionais *Unix*.

```
$ sudo apt-get install autoconf
```

4º Passo

Para a correta instalação do OMNET++, alguns pacotes são fundamentais para o funcionamento. Primeiramente, execute os comandos abaixo como administrador:

```
$ sudo apt-get install build-essential gcc g++ bison flex perl  
$ sudo apt-get install tcl-dev tk-dev libxml2-dev zlib1g-dev default-jre  
$ sudo apt-get install doxygen graphviz libwebkitgtk-1.0-0  
$ sudo apt-get install qt4-qmake libqt4-dev libqt4-opengl-dev
```

5º Passo (Opcional)

Para utilizar pacotes adicionais de visualização 3D do OMNET++ realize a instalação através do comando:

```
$ sudo apt-get install openscenegraph libopenscenegraph-dev  
$ sudo apt-get install openscenegraph-plugin-osgearth osgearth  
$ sudo apt-get install osgearth-data libosgearth-dev
```

6º Passo (Opcional)

A fim de realizar simulação paralela, será necessária a instalação de pacotes MPI:

```
$ sudo apt-get install openmpi-bin libopenmpi-dev
```

E.1.1 Instalação do SUMO

7º Passo

A instalação do SUMO é feita através do comando:

```
$ sudo apt-get install sumo sumo-tools sumo-doc
```

8º Passo (Opcional)

Para testar o funcionamento do SUMO, inicialize através do terminal:

```
§ sumo-gui
```

Vá direto para o 14º Passo.

E.1.2 Instalação do OMNET++

9º Passo

Descompacte o arquivo do OMNET++ obtido no 1º Passo. Após a descompactação da pasta, é necessário acessar o diretório raiz e estabelecer as variáveis de ambiente antes da configuração.

```
§ cd omnetpp-4.6  
§ . setenv
```

10º Passo

O terminal irá retornar o diretório atual indicando para onde as variáveis de ambiente do OMNET++ estão localizadas. Agora será necessário configurar a instalação previamente de acordo com as características do sistema operacional do usuário.

```
§ ./configure
```

Se a configuração for feita com sucesso, o sistema retornará uma mensagem similar à [Figura 66](#).

```
tcc@tcc-vb: ~/WAVE/omnetpp-5.0
checking for LibXML XML parser with CFLAGS="-O2 -DNDEBUG=1 -fPIC -fno-stack-protector -I/usr/include/libxml2" LIBS="-lxml2"... yes
checking for Expat XML parser with CFLAGS="-O2 -DNDEBUG=1 -fPIC -fno-stack-protector" LIBS="-lexpat"... yes
configure: Using LibXML for XML parsing
checking for zlib with CFLAGS="-O2 -DNDEBUG=1 -fPIC -fno-stack-protector" LIBS="-lz"... yes
checking for Akaroa with CFLAGS="-O2 -DNDEBUG=1 -fPIC -fno-stack-protector -I/usr/local/akaroa/include" LIBS="-L/usr/local/akaroa/lib -lakaroa -lfl"... no
configure: WARNING: Optional package Akaroa not found
configure: creating ./config.status
config.status: creating Makefile.inc
config.status: creating test/core/runtest

WARNING: The configuration script could not detect the following packages:
MPI (optional) PCAP (optional) Akaroa (optional)

Scroll up to see the warning messages (use shift+PgUp), and search config.log
for more details. While you can use OMNeT++ in the current configuration,
be aware that some functionality may be unavailable or incomplete.

Your PATH contains /home/tcc/WAVE/omnetpp-5.0/bin. Good!
tcc@tcc-vb:~/WAVE/omnetpp-5.0$
```

Figura 66 – Configuração bem sucedida do OMNET++ no Linux

Fonte: Autor

Se houver ausência de pacotes, o *script* de configuração informará o nome deste a ser instalado de forma similar à Figura 67.

```
tcc@tcc-vb: ~/WAVE/omnetpp-5.0
checking whether we are using the GNU C++ compiler... yes
checking whether g++ accepts -g... yes
checking for g++... g++
checking for ranlib... ranlib
checking whether g++ supports -std=c++11... yes
checking whether g++ supports -fno-stack-protector... yes
checking whether g++ supports -Wl,--no-as-needed... yes
checking whether g++ supports -Wl,--as-needed... yes
checking for swapcontext... yes
checking if shared libs need -fPIC... yes
checking for dlopen with CFLAGS="" LIBS=""... no
checking if --export-dynamic linker option is supported/needed... test failed
checking for flags needed to link with static libs containing simple modules...
-Wwhole-archive
configure: NOTE: Use the following syntax when linking with static libraries
configure: containing simple modules and other dynamically registered components :
configure:   g++ ... -Wl,--whole-archive <libs> -Wl,--no-whole-archive ...
configure: checking whether linker supports -rpath... yes
configure: checking for bison... no
configure: checking for byacc... no
configure: error: Bison not found, needed to build OMNeT++/OMNEST -- please install it!
tcc@tcc-vb:~/WAVE/omnetpp-5.0$
```

Figura 67 – Configuração mal sucedida do OMNET++ no Linux

Fonte: Autor

Retorne ao 4º Passo para certificar-se que todos os pacotes necessários foram corretamente instalados e refaça o 10º Passo.

11º Passo

Logo após a configuração, é necessário compilar o OMNET++ através do comando:

```
$ make
```

A execução deste comando levará algum tempo. Terminada a realização do comando, o OMNET++ terá sido instalado com sucesso.

12º Passo (Opcional)

Após a instalação, abra o OMNET++ pelo terminal através do comando:

```
$ omnetpp
```

Avance para o 17º Passo a fim de configurar o OMNET++.

E.2 Sistemas Windows

1º Passo

Após os arquivos compressos do SUMO, OMNET++ e VEINS terem sido obtidos em seus respectivos sites, organize os arquivos dentro da pasta padrão **WAVE**.

E.2.1 Instalação do SUMO

2º Passo

Descompacte o arquivo do SUMO obtido no 1º Passo. Assim que o processo terminar, acesse a pasta **bin**. Dentro desta pasta estão os executáveis do SUMO, como os programas úteis **netedit**, **netgenerate** e **netconvert**.

3º Passo (Opcional)

Clique no executável **sumo-gui** para abrir o *framework* SUMO. A fim de testar o SUMO, vá direto para o 14º Passo.

E.2.2 Instalação do OMNET++

4º Passo

O framework OMNET++ oferece suporte para os sistemas Windows 7 e 10 apenas. Certifique-se que o caminho completo do diretório do arquivo compactado do OMNET++ não contém nenhum espaço, como no exemplo: **C:\Users\tcc\WAVE\omnetpp-5.0-src-windows.zip**. Descompacte o arquivo no diretório padrão **WAVE** criado no obtido no 1º Passo. A pasta do OMNET++ contém um compilador C++, um ambiente de linha de comando e todos os arquivos requeridos para funcionamento do programa.

5º Passo

Acesse a pasta descompactada e execute como administrador o arquivo **mingwenv.cmd**. Este arquivo irá construir um terminal Linux para facilitar a configuração do OMNET++. Conforme a [Figura 68](#) orienta, pressione a tecla enter para descompactar os arquivos do terminal. Este processo pode durar alguns minutos.

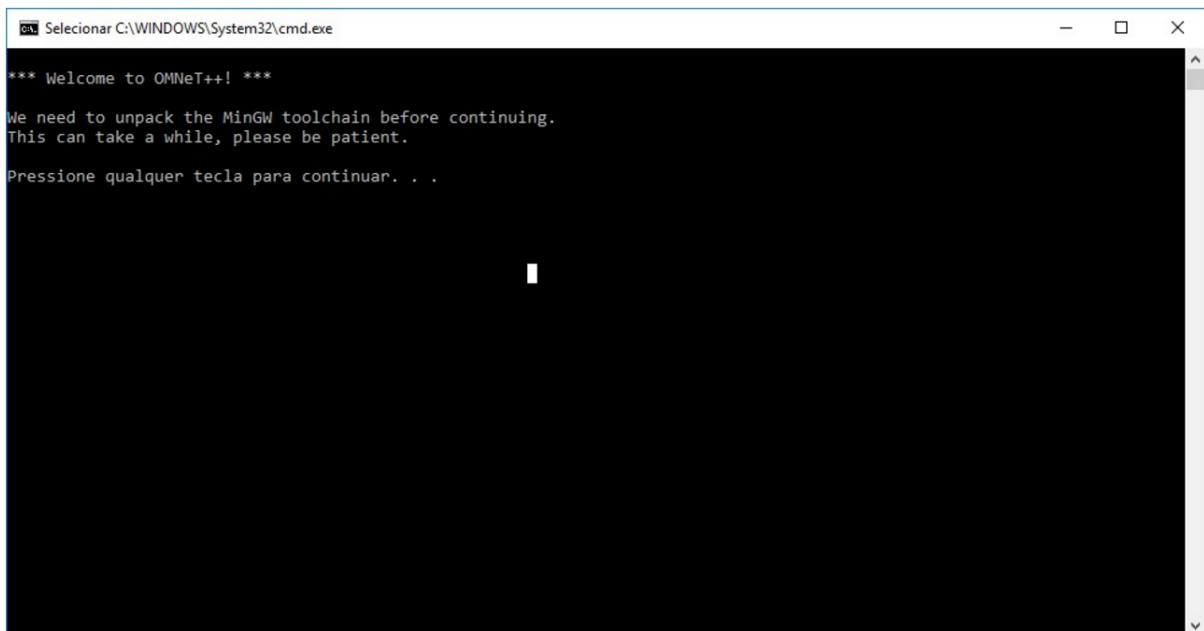


Figura 68 – Software mingwenv em execução no Windows

Fonte: Autor

Após o término da configuração, o arquivo **mingwenv.cmd** deverá mostrar a mensagem contida na [Figura 69](#).

```
Extracting win32\etc\pki\ca-trust\legacy
Extracting win32\etc\pki\ca-trust\extracted\pem
Extracting win32\etc\pki\ca-trust\extracted\openssl
Extracting win32\etc\pki\ca-trust\extracted\java
Extracting win32\etc\pki\ca-trust\extracted
Extracting win32\etc\pki\ca-trust
Extracting win32\etc\pkcs11
Extracting win32\etc\pacman.d\gnupg
Extracting win32\etc\pacman.d
Extracting win32\etc\krb5
Extracting win32\etc\fstab.d
Extracting win32\etc
Extracting win32\dev\shm
Extracting win32\dev\mqueue
Extracting win32\dev
Extracting win32

Everything is Ok

Folders: 2110
Files: 28954
Size: 1815134150
Compressed: 261199179

QtBinPatcher v2.2.0. Tool for patching paths in Qt binaries.
Yuri V. Krugloff, 2013-2015. http://www.tver-soft.org
This is free software released into the public domain.
```

Figura 69 – Configuração do Terminal Virtual no Windows

Fonte: Autor

6º Passo

Aperte a tecla enter e o terminal virtual Linux será mostrado conforme a Figura 70 abaixo.

```
Welcome to OMNeT++ 5.0!
Type "./configure" and "make" to build the simulation libraries.
When done, type "omnetpp" to start the IDE.

/c/Users/vitor/Desktop/WAVE/omnetpp-5.0$ |
```

Figura 70 – Terminal Virtual no Windows

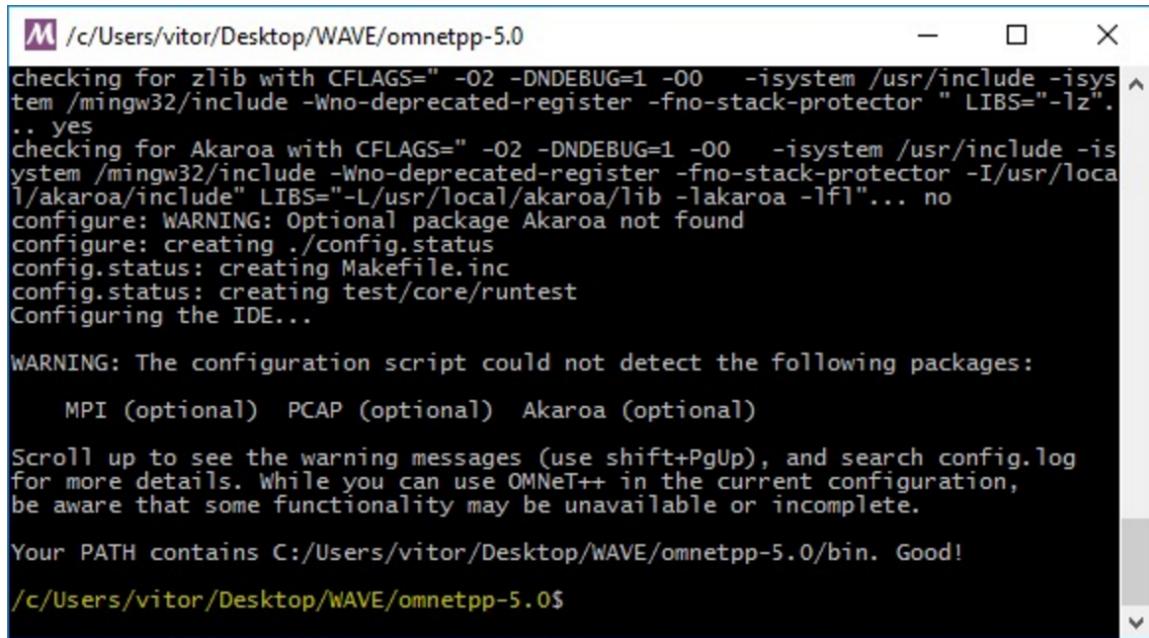
Fonte: Autor

7º Passo

Prossiga com a instalação, executando o comando para pré-configurar o sistema.

```
$ ./configure
```

A pré-configuração do sistema pode levar alguns minutos. Após finalizada, uma mensagem será mostrada conforme a [Figura 71](#).



The screenshot shows a terminal window with the following text output:

```
M /c/Users/vitor/Desktop/WAVE/omnetpp-5.0
checking for zlib with CFLAGS="-O2 -DNDEBUG=1 -O0 -isystem /usr/include -isystem /mingw32/include -Wno-deprecated-register -fno-stack-protector" LIBS="-lz"...
.. yes
checking for Akaroa with CFLAGS="-O2 -DNDEBUG=1 -O0 -isystem /usr/include -isystem /mingw32/include -Wno-deprecated-register -fno-stack-protector -I/usr/local/akaroa/include" LIBS="-L/usr/local/akaroa/lib -lakaroa -lfl"... no
configure: WARNING: Optional package Akaroa not found
configure: creating ./config.status
config.status: creating Makefile.inc
config.status: creating test/core/runttest
Configuring the IDE...

WARNING: The configuration script could not detect the following packages:
          MPI (optional)  PCAP (optional)  Akaroa (optional)

Scroll up to see the warning messages (use shift+PgUp), and search config.log
for more details. While you can use OMNeT++ in the current configuration,
be aware that some functionality may be unavailable or incomplete.

Your PATH contains C:/Users/vitor/Desktop/WAVE/omnetpp-5.0/bin. Good!
/c/Users/vitor/Desktop/WAVE/omnetpp-5.0$
```

Figura 71 – Configuração bem sucedida do OMNET++ no Windows

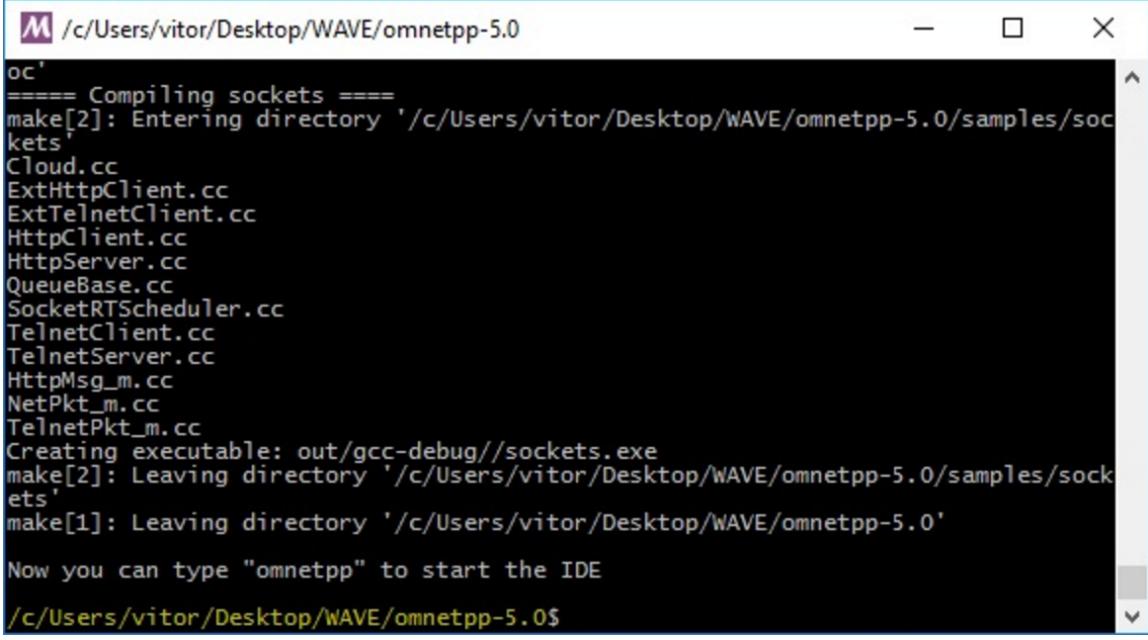
Fonte: Autor

8º Passo

Execute o comando que irá construir as bibliotecas.

```
$ make
```

A configuração das bibliotecas do OMNET++ no sistema é demorada. Assim que for finalizada, uma mensagem será mostrada no terminal virtual como pode ser visto na [Figura 72](#).



The screenshot shows a terminal window titled 'M /c/Users/vitor/Desktop/WAVE/omnetpp-5.0'. The window displays the output of a make command for the 'sockets' module. The output includes:

```
oc'
===== Compiling sockets =====
make[2]: Entering directory '/c/Users/vitor/Desktop/WAVE/omnetpp-5.0/samples/sockets'
Cloud.cc
ExtHttpClient.cc
ExtTelnetClient.cc
HttpClient.cc
HttpServer.cc
QueueBase.cc
SocketRTScheduler.cc
TelnetClient.cc
TelnetServer.cc
HttpMsg_m.cc
NetPkt_m.cc
TelnetPkt_m.cc
Creating executable: out/gcc-debug//sockets.exe
make[2]: Leaving directory '/c/Users/vitor/Desktop/WAVE/omnetpp-5.0/samples/sockets'
make[1]: Leaving directory '/c/Users/vitor/Desktop/WAVE/omnetpp-5.0'

Now you can type "omnetpp" to start the IDE
/c/Users/vitor/Desktop/WAVE/omnetpp-5.0$
```

Figura 72 – Instalação bem sucedida do OMNET++ no Windows

Fonte: Autor

9º Passo (Opcional)

No fim da instalação execute o comando abaixo no terminal virtual para executar o *framework*.

```
$ omnetpp
```

A fim de configurar o OMNET++, avance para o 17º Passo.

E.3 Testar funcionamento do SUMO

14º Passo (Opcional)

Abra o exemplo test.sumo.cfg dentro da pasta do SUMO no caminho .../sumo-0.25.0/tools/contributed/traci4j/test/resources/sumo_maps/box11 através do menu *File > Open Simulation*.

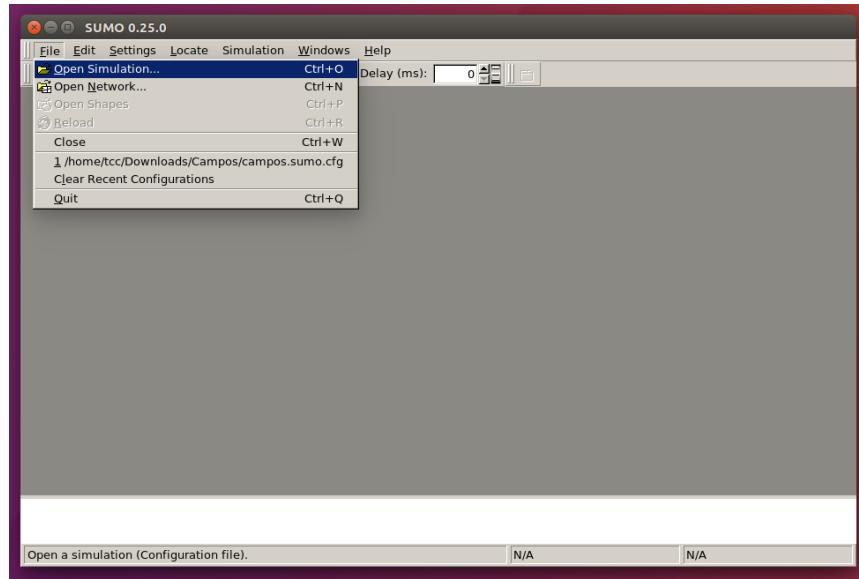


Figura 73 – Abrindo uma Simulação no SUMO

Fonte: Autor

15º Passo (Opcional)

Haverá uma simulação de tráfego rodoviário. Para tornar a simulação mais perceptível, aumente o *delay*. Também é possível mudar o esquema de apresentação do mapa, alterando de *standard* para *real world*. Para inicializar a simulação, clique no botão indicado pelo círculo vermelho na [Figura 74](#) (ou pressione o atalho Ctrl + A).

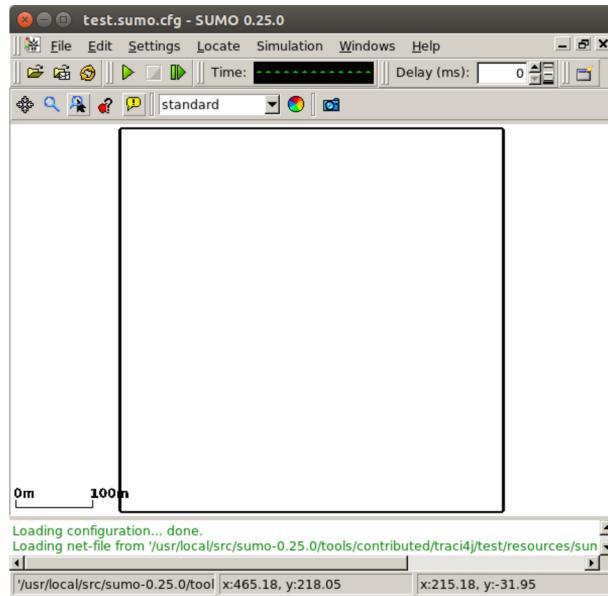


Figura 74 – Inicializando uma Simulação no SUMO

Fonte: Autor

16º Passo (Opcional)

Os veículos vão surgir em cada esquina e se movimentarão indicando que o teste foi realizado com sucesso.

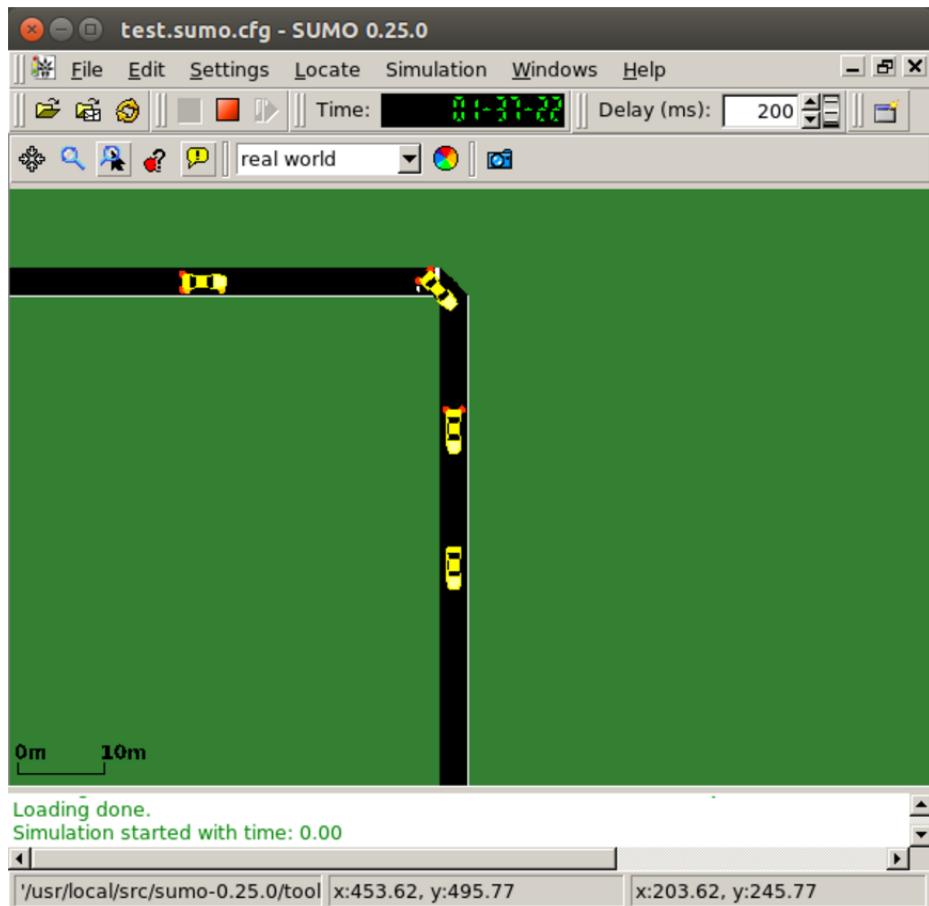


Figura 75 – Visualizando uma Simulação no SUMO

Fonte: Autor

E.4 Configuração do OMNET++

17º Passo

Ao executar o OMNET++, é necessário definir um *workspace*. É recomendado manter o *workspace* pré-definido pelo OMNET++.

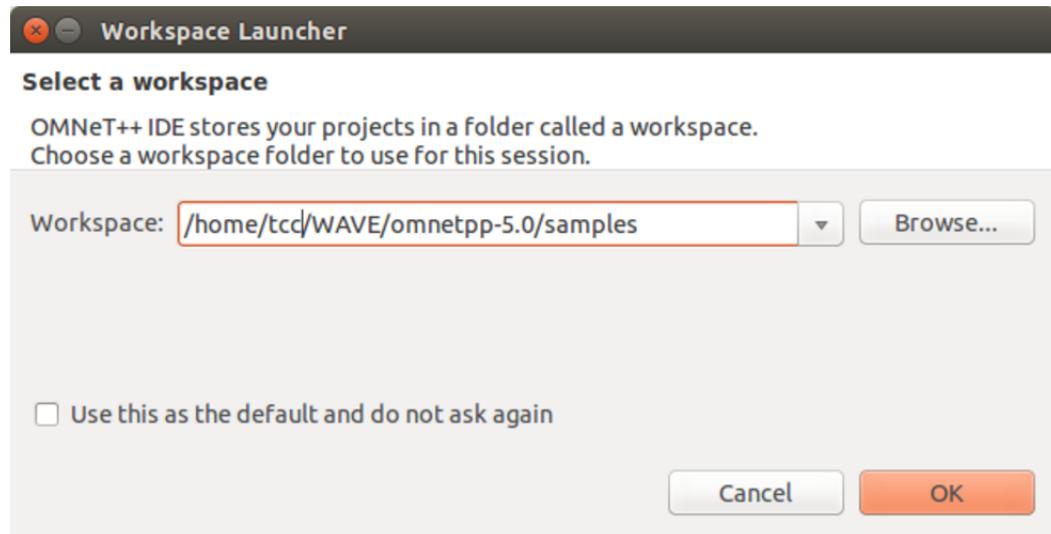


Figura 76 – Executando o OMNET++

Fonte: Autor

A tela de boas vindas do OMNET++ é mostrada. Conseguimos identificar semelhanças com a IDE do Eclipse.



Figura 77 – Tela de Boas vindas do OMNET++

Fonte: Autor

18º Passo (Opcional)

Se o OMNET++ solicitar a instalação do projeto INET e a importação de exemplos, clique em cancelar.

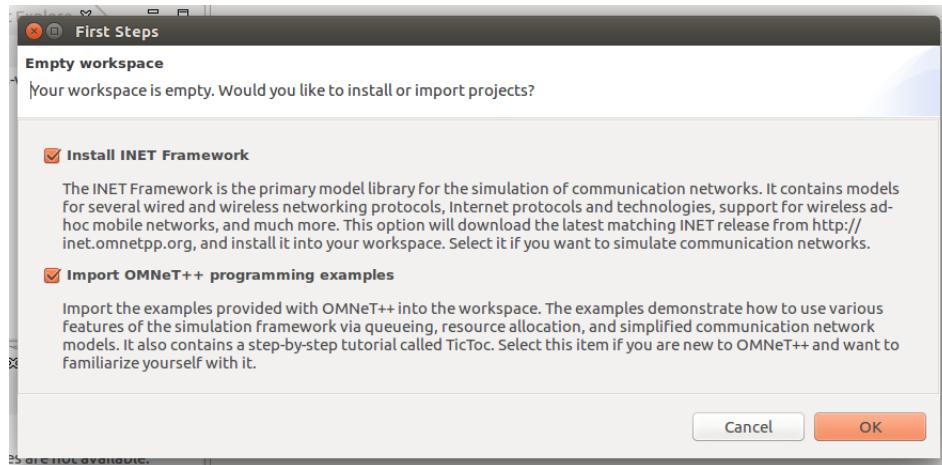


Figura 78 – Importação de Exemplos

Fonte: Autor

E.5 Importação do VEINS

O VEINS é um projeto que funciona dentro do OMNET++, portanto há de ser importado. Os procedimentos de importação ocorrem de forma igual nos sistemas operacionais tratados neste tutorial.

19º Passo

Descompacte o arquivo do VEINS baixado na pasta padrão **WAVE**, criada no 1º Passo, e acesse o menu *File > Import > General: Existing Projects into Workspace* para importar o projeto.

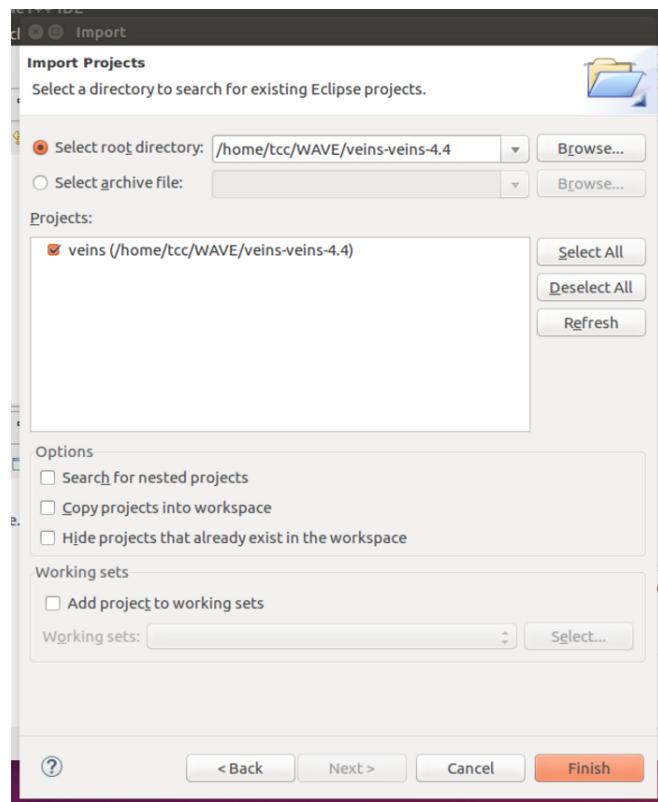


Figura 79 – Importação do VEINS

Fonte: Autor

O projeto VEINS será listado no *Project Explorer*.

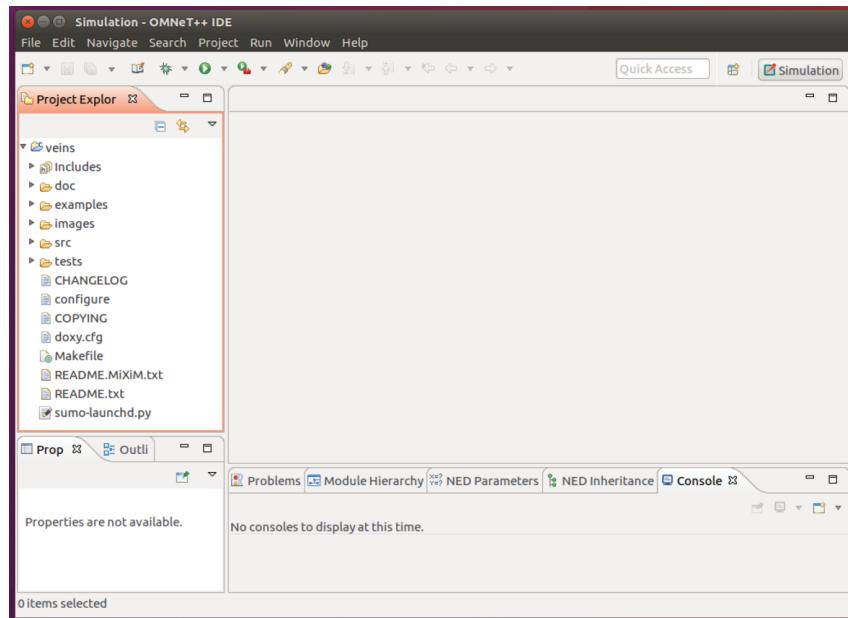


Figura 80 – VEINS importado no OMNET++

Fonte: Autor

20º Passo (Opcional)

É preciso construir o projeto VEINS importado acessando o menu *Project > Build All*. A construção pode levar alguns minutos.

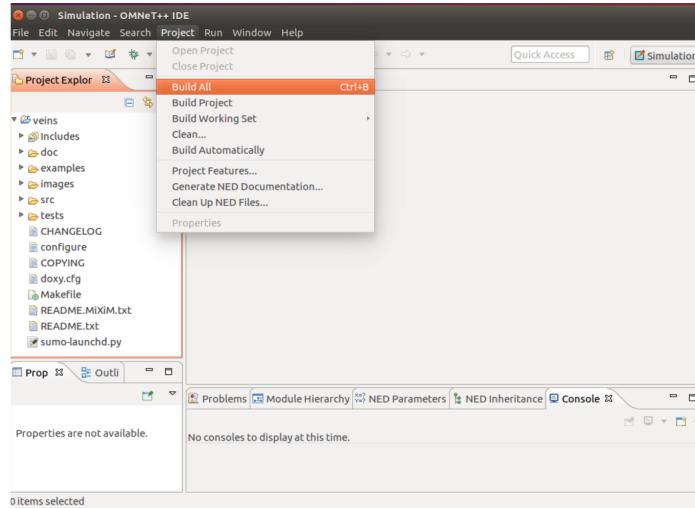


Figura 81 – Construindo o VEINS no OMNET++

Fonte: Autor

Certifique-se de o VEINS foi construído sem erros. No *Console*, aparecerá a mensagens *Build Finished* na última linha.

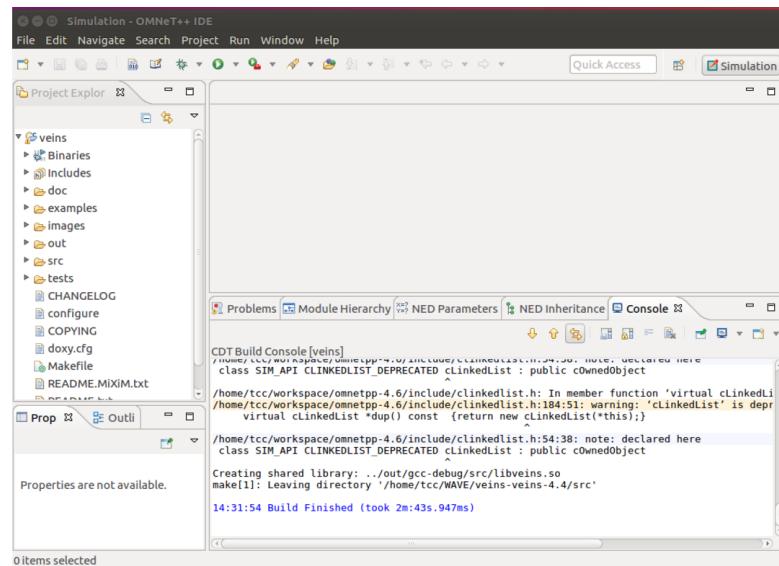


Figura 82 – VEINS Construído com Sucesso

Fonte: Autor

E.5.1 Testar integração dos *Frameworks*

21º Passo

No *Project Explorer* do OMNET++, acesse a pasta *veins > examples > veins*. Nela há um exemplo de simulação desenvolvida pelo criador do *framework* VEINS.

22º Passo

Para integrar os *frameworks* OMNET++ e SUMO, é necessário utilizar um *daemon* que serve para executar processos em plano de fundo. O *daemon* é disponibilizado no diretório do VEINS através do arquivo python **sumo-launchd.py**. Ele executará a interface gráfica do SUMO e trocará mensagens de *status* com o projeto VEINS do OMNET++.

Nos sistemas Linux, através do terminal acesse o diretório raiz da pasta VEINS e execute o seguinte comando:

```
$ ./sumo-launchd.py -vv -c sumo-gui
```

```
tcc@tcc-vb:~/WAVE/veins-veins-4.4
tcc@tcc-vb:~$ ls
Desktop Downloads Music Public Videos workspace
Documents examples.desktop Pictures Templates WAVE
tcc@tcc-vb:~$ cd WAVE/
tcc@tcc-vb:~/WAVE$ ls
omnetpp-4.6 veins-veins-4.4
tcc@tcc-vb:~/WAVE$ cd veins-veins-4.4/
tcc@tcc-vb:~/WAVE/veins-veins-4.4$ ls
CHANGELOG doc images README.MiXiM.txt sumo-launchd.py
configure doxy.cfg Makefile README.txt tests
COPYING examples out src
tcc@tcc-vb:~/WAVE/veins-veins-4.4$ ./sumo-launchd.py -vv -c sumo-gui
Logging to /tmp/sumo-launchd.log
Listening on port 9999
```

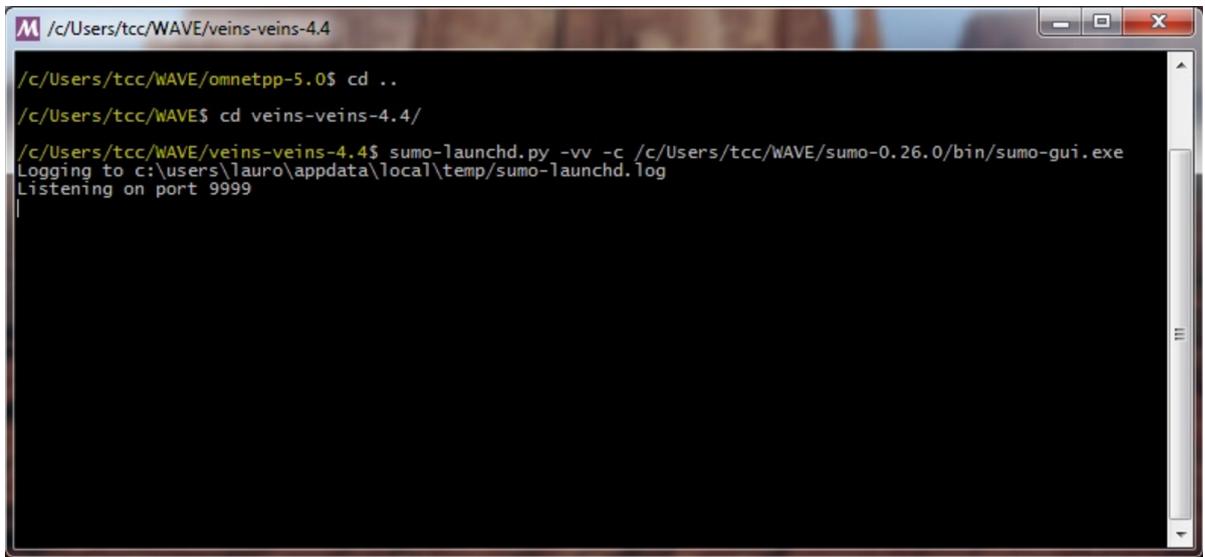
Figura 83 – Executando o *daemon* no Linux

Fonte: Autor

Nos sistemas Windows, acesse o diretório raiz do VEINS através do terminal virtual **mingwenv.cmd** obtido no 5º Passo da instalação Windows. Execute o *daemon* seguido da pasta onde se encontra o executável *sumo-gui* do SUMO. Atenção, o comando a seguir é apenas um exemplo. Modifique a parte em negrito apontando para o diretório do SUMO em seu computador.

```
$ sumo-launchd.py -vv -c /c/Users/tcc/WAVE/sumo-0.26.0/bin/sumo-gui.exe
```

O script executado irá realizar um *proxy* entre as mensagens do OMNET++ e do SUMO na porta 9999, o que é indicado pela mensagem no terminal *Listening on port 9999*.



```
/c/Users/tcc/WAVE/omnetpp-5.0$ cd ..
/c/Users/tcc/WAVE$ cd veins-veins-4.4/
/c/Users/tcc/WAVE/veins-veins-4.4$ sumo-launchd.py -vv -c /c/Users/tcc/WAVE/sumo-0.26.0/bin/sumo-gui.exe
Logging to c:\users\lauro\appdata\local\temp\sumo-launchd.log
Listening on port 9999
```

Figura 84 – Executando o *daemon* no Windows

Fonte: Autor

23º Passo

Deixe a janela do terminal aberta e volte à IDE do OMNET++. No *Project Explorer*, acesse a pasta *veins* > *examples* > *veins*. Nela há um exemplo de simulação desenvolvida pelo criador do framework VEINS. Clique com o botão direito sobre o arquivo de configuração **omnetpp.ini** e escolha a opção *Run As* > *OMNeT++ simulation*.

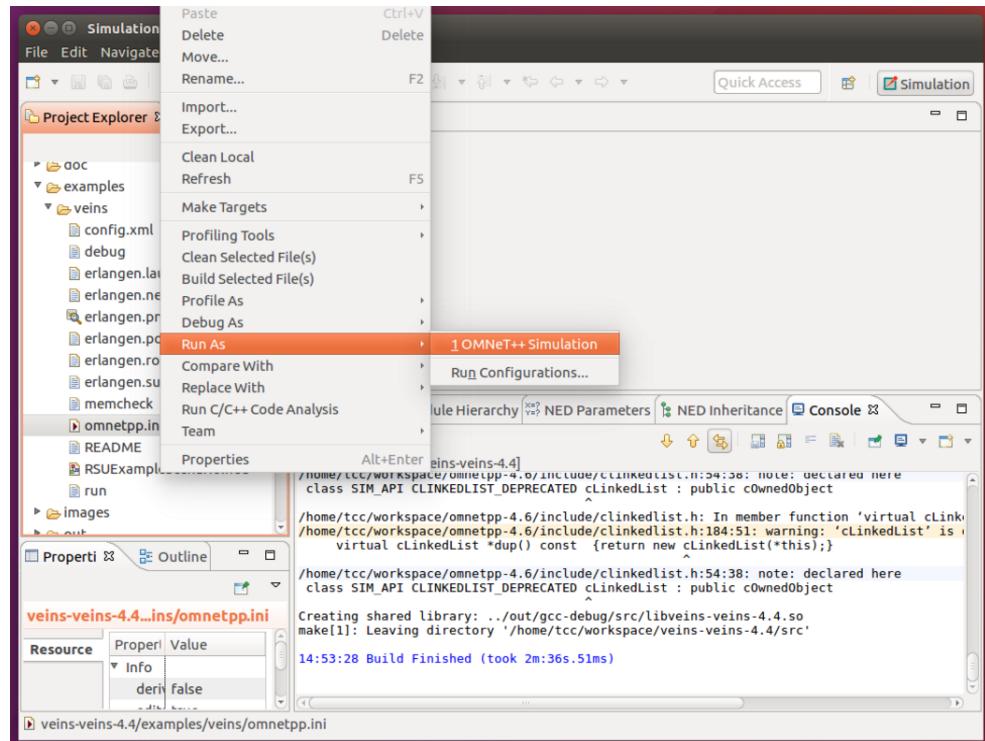


Figura 85 – Inicializando a Simulação Teste

Fonte: Autor

Uma caixa de diálogo indicará que uma configuração de execução foi criada. Apenas clique em *OK* para prosseguir.

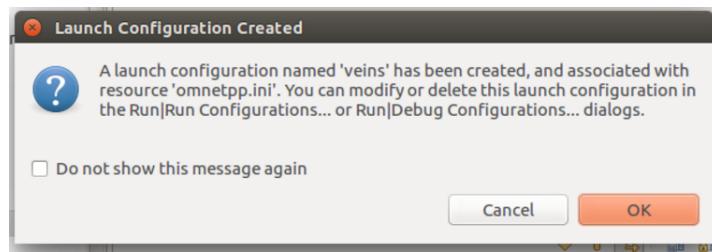


Figura 86 – Configuração de Execução Criada

Fonte: Autor

24º Passo

Uma tela da interface gráfica para execução de simulações (Tkenv) será aberta. Na caixa de diálogo, selecione em *Config name* a opção *debug*. Clique no ícone *Run*.

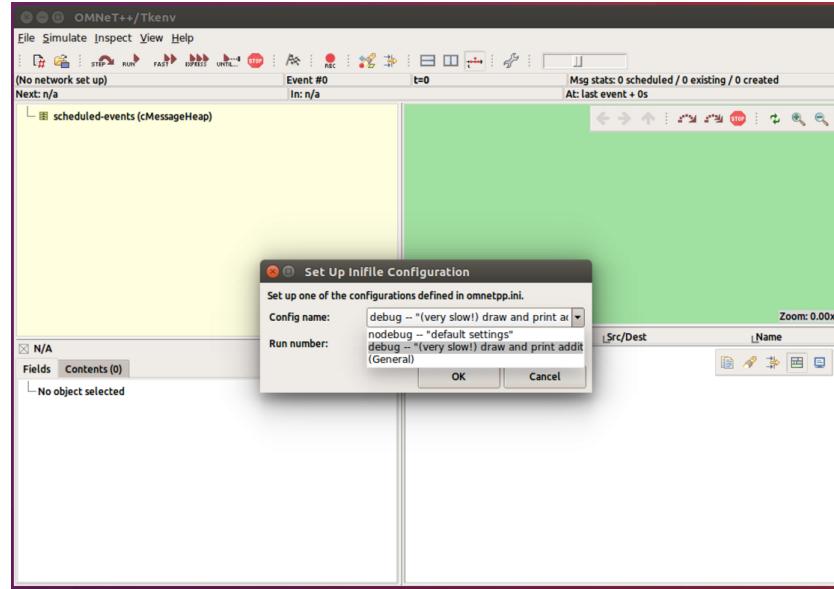


Figura 87 – Executando a Simulação Teste

Fonte: Autor

25º Passo (Opcional)

O *sumo-gui* será inicializado com o mapa da simulação já importado. Os dois softwares irão executar em conjunto, comunicando-se através do *daemon* inicializado alguns passos atrás.

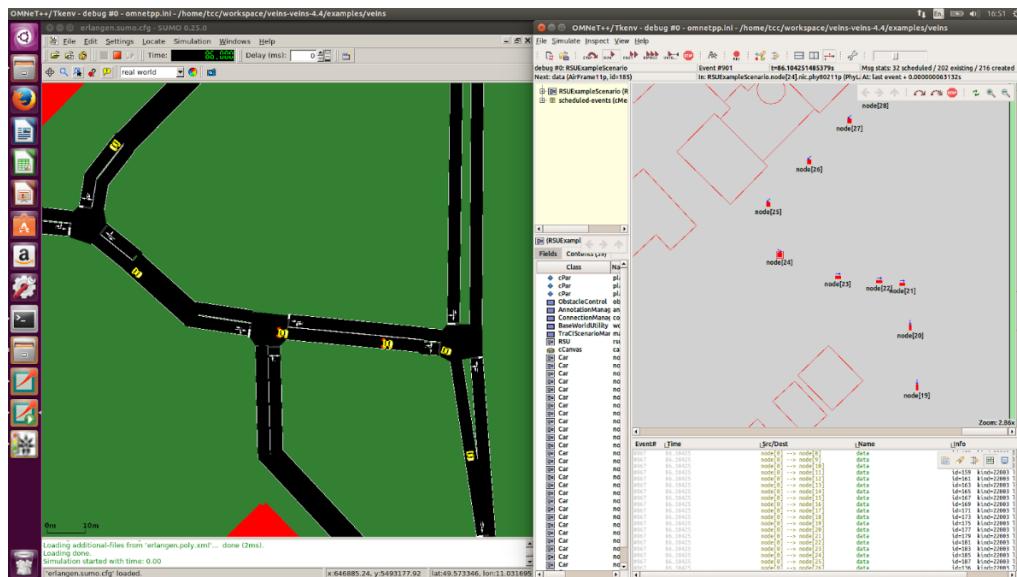


Figura 88 – Simulação Teste em Execução (SUMO e Tkenv)

Fonte: Autor

Neste cenário, os carros trocam informações WAVE (*Airframes*) de orientação de rota a partir de um acidente na pista. Unidades de acostamento (RSU ou *Roadside Units*)

e unidades de bordo de veículos (OBUs ou *Onboard Units*) transferem estas informações a partir do momento do acidente.

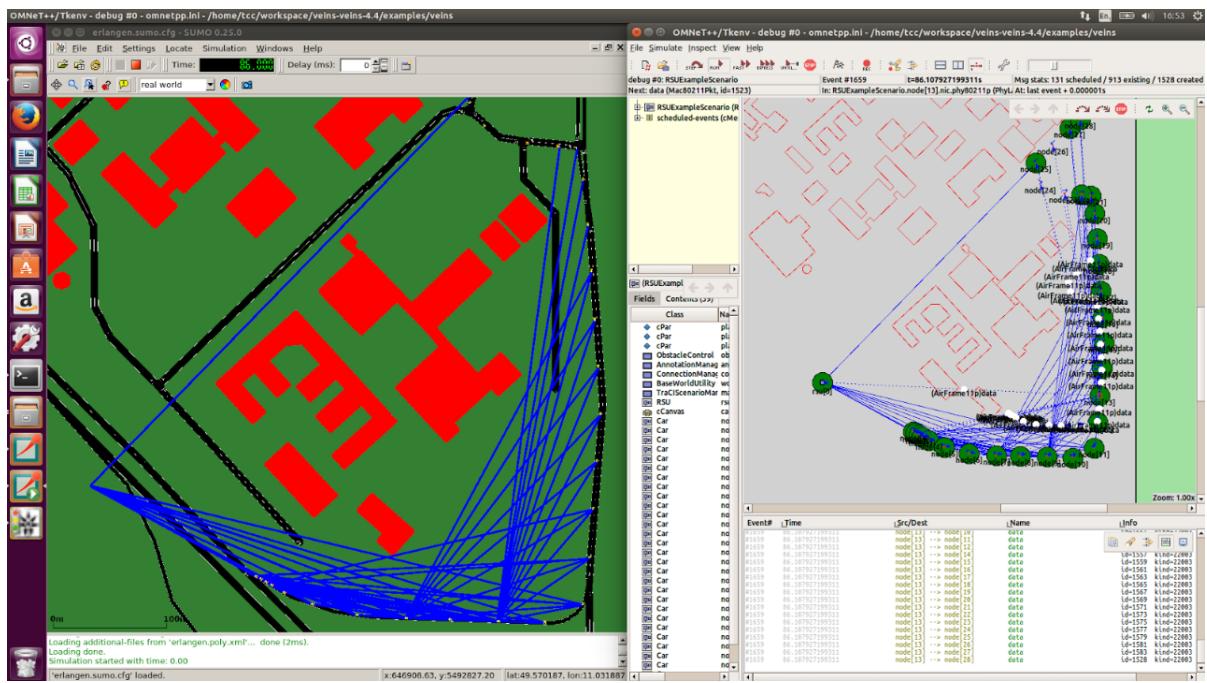


Figura 89 – Troca de Mensagens entre os Veículos

Fonte: Autor

ANEXO F – Desenvolvendo Simulações

Para o desenvolvimento de uma nova simulação, é necessário que os passos do Apêndice E tenham sido executados com sucesso.

1º Passo

Dentro do diretório padrão **WAVE** (1º Passo do Apêndice E), crie uma nova pasta com o nome que desejar. Para este tutorial, o nome da pasta será **simulacao**.

F.1 Exportando mapas através do OpenStreetMap

OpenStreetMap é um projeto colaborativo para criação do mapa mundial que seja editável e livre.

2º Passo

Acesse o link <https://www.openstreetmap.org/> e pesquise o endereço do local desejado.

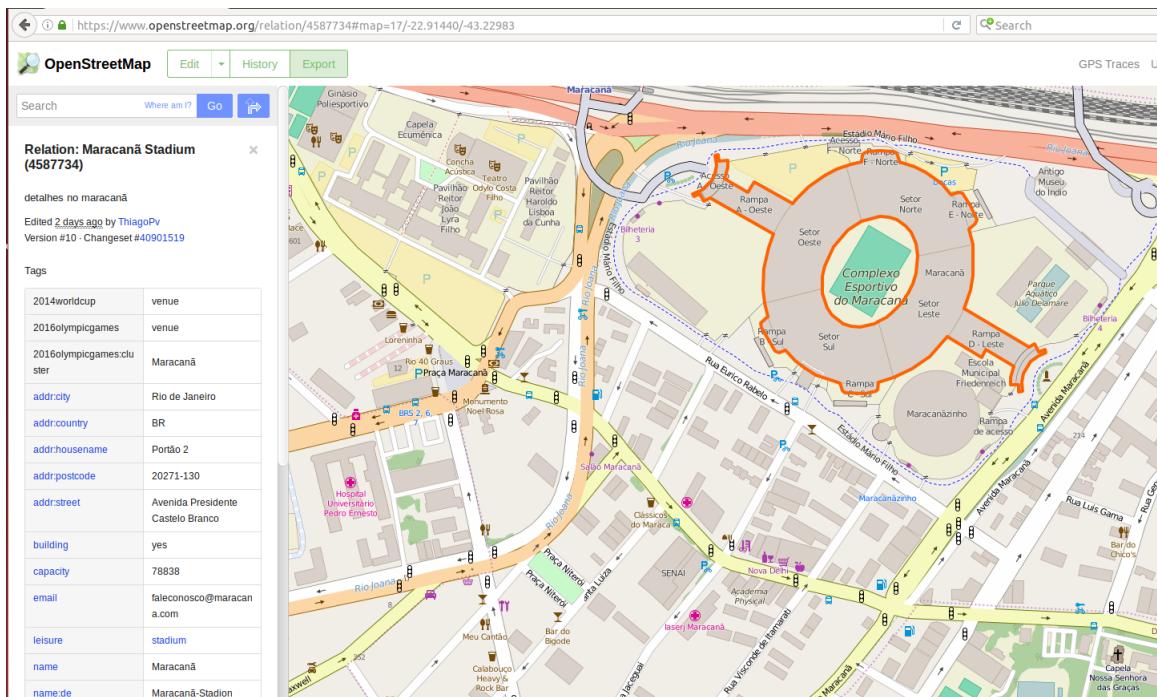


Figura 90 – Pesquisando um Endereço no OpenStreetMap

Fonte: Autor

3º Passo

Clique no botão *Export* (verde) para habilitar a exportação do mapa.

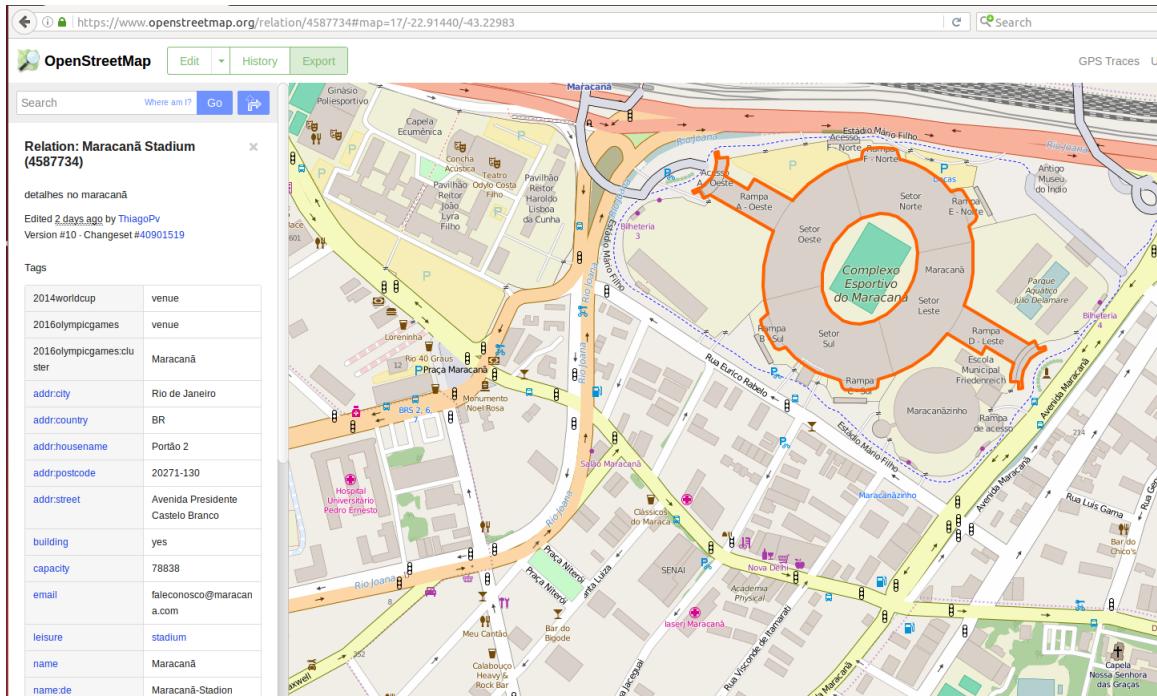


Figura 91 – Habilitando Exportação de Mapas no OpenStreetMap

Fonte: Autor

4º Passo (Opcional)

Clique no link *Manually select a different area* para selecionar uma área específica do mapa.

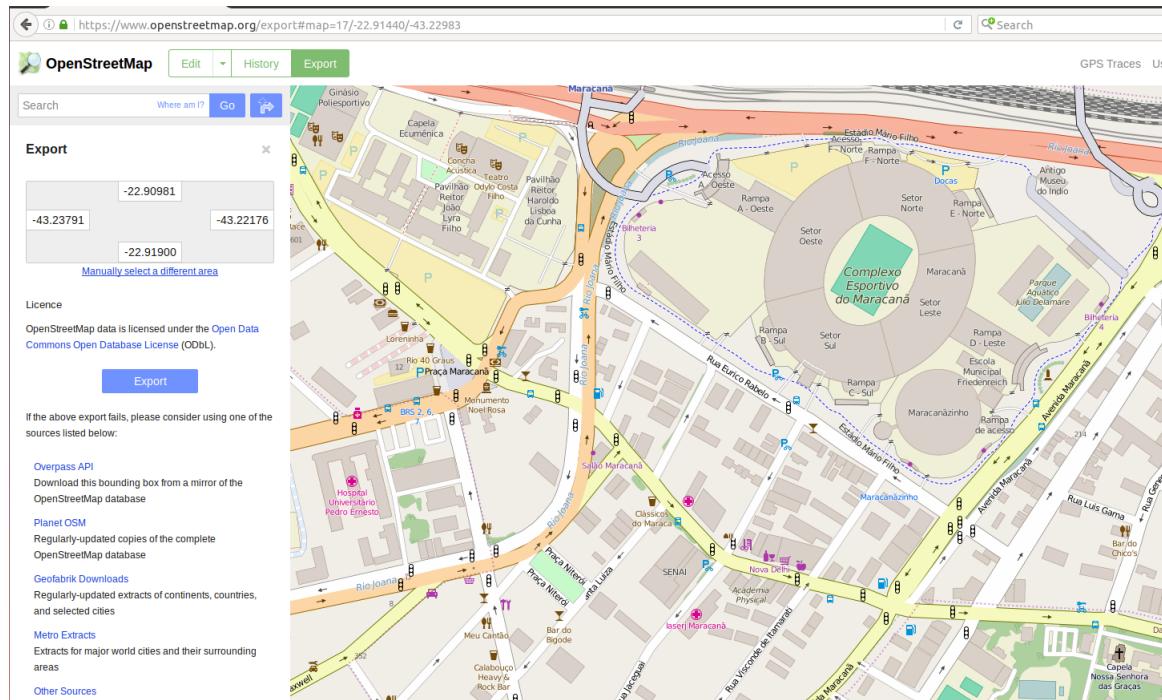


Figura 92 – Habilitando Seleção de Área no OpenStreetMap

Fonte: Autor

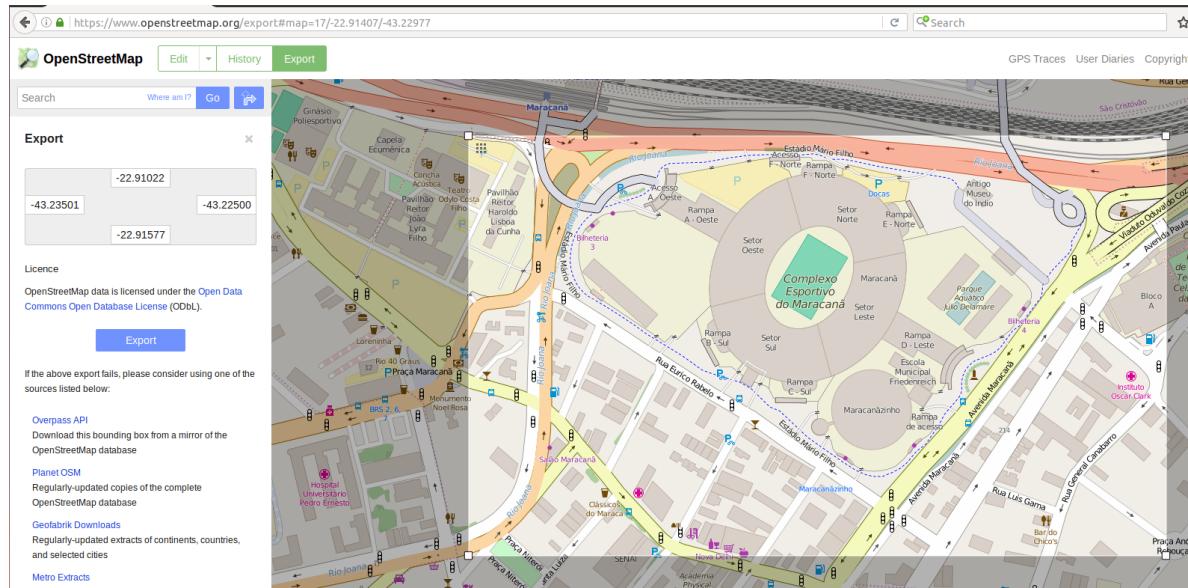


Figura 93 – Selecionando Área para Exportar no OpenStreetMap

Fonte: Autor

5º Passo

Clique no segundo botão *Export* (azul) e salve o arquivo na pasta **simulacao** (criada no 1º Passo do [Apêndice F](#)).

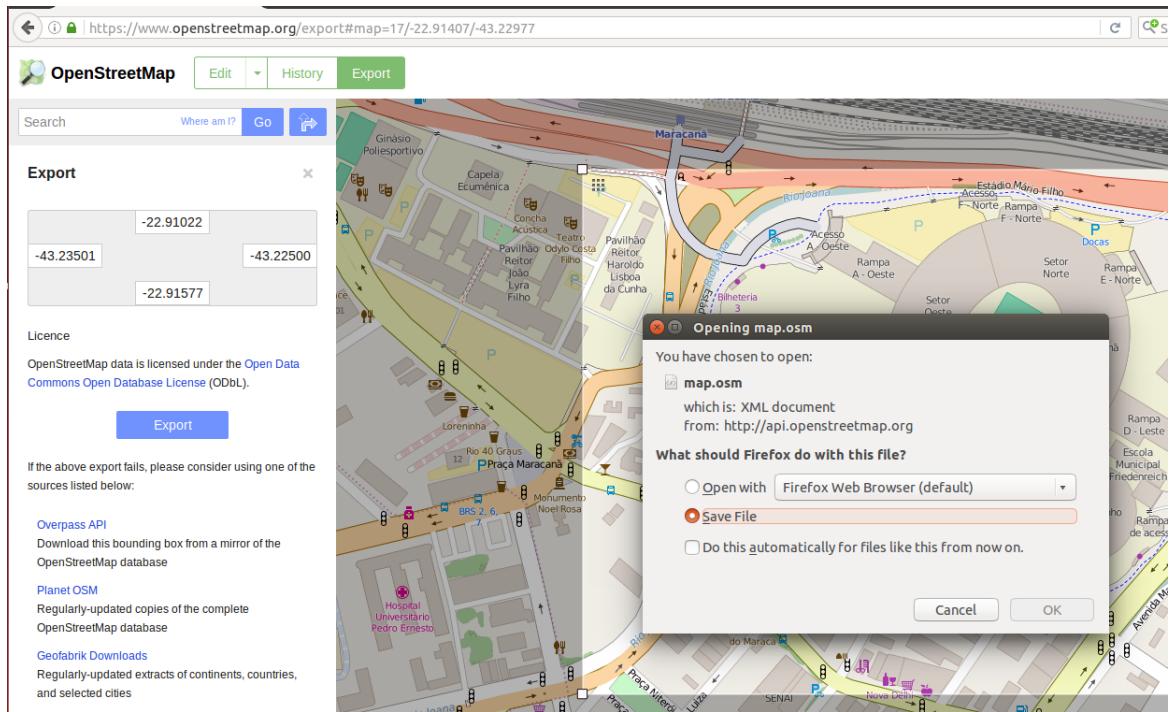


Figura 94 – Exportando Mapa no OpenStreetMap

Fonte: Autor

6º Passo

Abra o terminal linux (ou o terminal virtual **mingwenv.cmd** obtido no 5º Passo da instalação Windows no [Apêndice E](#)) e vá até o diretório onde encontra-se o arquivo **map.osm** obtido e execute o seguinte comando:

```
$ netconvert --osm-files map.osm -o maracana.net.xml
```

Atenção: mude o nome do mapa para atender à suas necessidades. O mapa foi convertido de **map.osm** para **maracana.net.xml** apenas como exemplo.

F.2 Criando polígonos para o mapa

É possível transferir os prédios do OpenStreetMap para o SUMO. Estes prédios poderão ser utilizados como obstáculos causando atenuação dos sinais para uma simulação mais realista.

7º Passo

Procure pelo arquivo **osmPolyconvert.typ.xml** dentro da pasta do SUMO e copie este arquivo para a pasta **simulacao** (criada no 1º Passo do [Apêndice F](#)).

Para procurar este arquivo no linux, utilize o comando **locate osmPolyconvert** através do terminal. Para procurar no windows, va até o diretório do SUMO extraído na pasta **WAVE** e pesquise através do *explorer*.

8º Passo

Execute a linha de comando a seguir, lembrando de adaptá-la:

```
$ polyconvert --net-file maracana.net.xml --osm-files map.osm --type-file
osmPolyconvert.typ.xml -o maracana.poly.xml
```

F.3 Gerando rotas aleatórias

Com o mapa pronto, é possível incluir veículos e gerar rotas aleatórias utilizando uma das ferramentas do SUMO.

9º Passo

No diretório do SUMO, procure pelo arquivo **randomTrips.py**. Este arquivo provavelmente estará dentro da pasta **tools**.

```
$ locate randomTrips.py
```

10º Passo

Copie os seguinte três itens abaixo para a pasta **simulacao** (criada no 1º Passo do [Apêndice F](#)):

randomTrips.py	Programa principal de geração de rotas aleatórias.
route2trips.py	Programa de apoio ao randomTrips.py .
sumolib	Pasta contendo diversas bibliotecas para a execução do randomTrips.py .

11º Passo

No diretório **simulacao** (10º Passo do [Apêndice F](#)), adapte a linha de comando abaixo e execute-a no terminal:

```
$ python randomTrips.py -n maracana.net.xml -r maracana.rou.xml -e 50 -l
```

F.4 Preparando a simulação

12º Passo

Copie o arquivo de configuração exemplo do VEINS localizado no diretório `veins-veins-4.4 > examples > veins > erlangen.sumo.cfg` para a pasta `simulacao`. Altere o nome deste arquivo para o mesmo padrão utilizado nos outros arquivos. Exemplo: `maracana.sumo.cfg`.

13º Passo

Abra o arquivo `maracana.sumo.cfg` com um editor de texto e mude os valores do `input` para ficar de acordo com os arquivos gerados nos passos anteriores.

```
<input>
    <net-file value="maracana.net.xml"/>
    <route-files value="maracana.rou.xml"/>
    <additional-files value="maracana.poly.xml"/>
</input>
```

14º Passo (Opcional)

O SUMO verifica online os arquivos xml passados como parâmetros. Para evitar a verificação (e agilizar a execução da simulação), adicione o seguinte após o `input`:

```
</input>

<report>
    <xml-validation value="never"/>
</report>
```

15º Passo (Opcional)

Abra o SUMO, aumente o `delay` e execute a simulação novamente através do menu `File > Reload`.

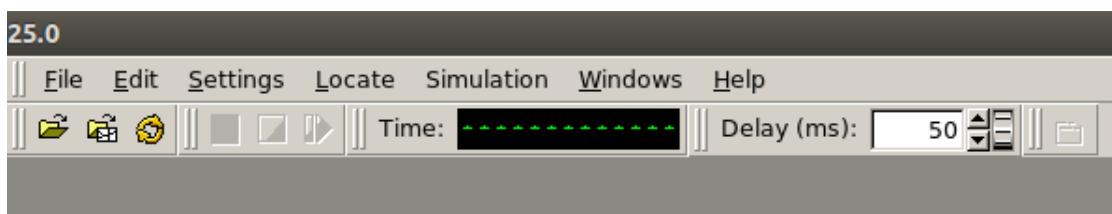


Figura 95 – Alterando o *Delay*

Fonte: Autor

Carregue o arquivo **maracana.sumo.cfg** através do menu *File > Open Simulation...*

```
§ sumo-gui -c maracana.sumo.cfg
```

F.5 Simulando a Rede Veicular

16º Passo

Copie a pasta **simulacao** (criada no 1º Passo do [Apêndice F](#)) para o diretório *veins-veins-4.4 > examples*.

17º Passo

Copie os arquivos **omnetpp.ini**, **RSUExampleScenario.ned** e **erlangen.launchd.xml** localizados no diretório *veins-veins-4.4 > examples > veins* para a pasta *veins-veins-4.4 > examples > simulacao*.

18º Passo

Altere o nome do **RSUExampleScenario.ned** e do **erlangen.launchd.xml** para seguir o mesmo modelo dos outros arquivos.

```
MaracanaScenario.ned e maracana.launchd.xml.
```

19º Passo

Abra o arquivo **maracana.launchd.xml** e altere o nome dos arquivos para adequar ao cenário:

```
<?xml version="1.0"?>
<!-- debug config -->
<launch>
    <copy file="maracana.net.xml"/>
    <copy file="maracana.rou.xml"/>
    <copy file="maracana.poly.xml"/>
    <copy file="maracana.sumo.cfg" type="config"/>
</launch>
```

20º Passo

Abra o arquivo **MaracanaScenario.ned** e altere o nome da *network* para o mesmo nome do arquivo:

```
network MaracanaScenario extends Scenario
```

21º Passo

Abra o arquivo **omnetpp.ini**, procure as duas linhas abaixo:

```
network = RSUExampleScenario
*.manager.launchConfig = xmldoc("erlangen.launchd.xml")
```

Altere estas linhas para adequá-las ao cenário:

```
network = MaracanaScenario
*.manager.launchConfig = xmldoc("maracana.launchd.xml")
```

22º Passo

Execute o OMNET++, clique com o botão direito em cima do projeto **veins** e clique em **Properties**. Na janela que surgir, digite **NED** e clique em **NED Source Folders**.

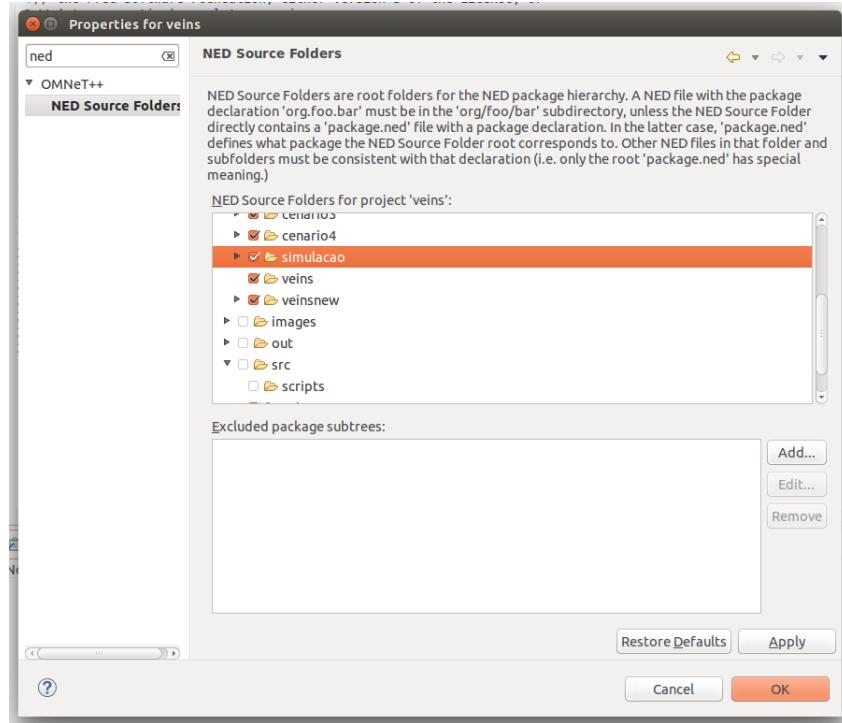


Figura 96 – Configurando NED para a nova Simulação

Fonte: Autor

Selecione a pasta **simulacao** (criada no 1º Passo do [Apêndice F](#)) e clique em **OK**.

22º Passo

Na aba **Project Explorer**, navegue até a pasta **simulacao** e siga o resto do tutorial a partir do 22º Passo do [Apêndice E](#) adaptando-o para a nova simulação.

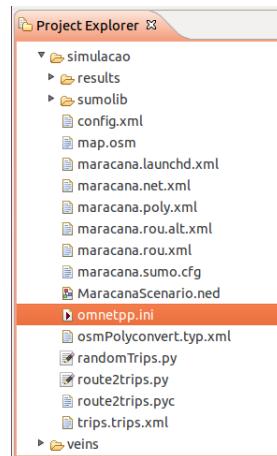


Figura 97 – Nova Simulação no *Project Explorer*

Fonte: Autor

ANEXO G – Tabelas e Código Fonte

Este anexo contém os endereços eletrônicos para visualização das tabelas e código fonte da aplicação de emergência do Estudo de Caso deste trabalho citadas ao longo da seção 5.2.

- **Testes com periodicidade e número de ruas** em <https://goo.gl/pfvQzr>;
- **Experimentos 1 e 2** em <https://goo.gl/X6ed11>;
- **Experimento 3** em <https://goo.gl/ewVL6u>;
- **Código Fonte** em <https://github.com/laurodelacerda/veins>.