

Departamento de Sistemas e Computação – FURB  
Curso de Ciência da Computação  
Trabalho de Conclusão de Curso – 2017/2

# APLICAÇÃO PARA MONITORAMENTO VEICULAR EM TEMPO REAL

Acadêmico  
Maicon Machado Gerardi da Silva  
maicon.gerardi@gmail.com

Orientador  
Prof. Miguel Alexandre Wisintainer  
maw@furb.br

# Roteiro

- Motivação
- Objetivo Geral
- Objetivos Específicos
- Fundamentação Teórica
- Trabalhos Correlatos
- Ferramentas Atuais
- Especificação
- Implementação
- Operacionalidade da Implementação
- Análise dos Resultados
- Conclusões
- Sugestões
- Demonstração

# Motivação

- Alto índice de furtos a veículos
- Falhas mecânicas causadas por falta de manutenção
- Integrar as soluções de Staroski (2016) e Baumgarten (2016) em uma única plataforma
- Explorar os recursos de uma nova placa
- Principais interesses de aprendizado:
  - Linguagem de programação Python
  - Internet da Coisas (IoT)
  - Hardware
  - Redes

# Objetivo Geral

Construir uma aplicação que abrange desenvolver um **software embarcado** e um **aplicativo *mobile***.

Software Embarcado: coletar a posição geográfica, imagens e dados da porta OnBoard Diagnostic (OBD) de um automóvel.

Aplicativo *Mobile*: capturar e disponibilizar as informações desse software embarcado.

# Objetivos Específicos

- Integrar a placa Raspberry Pi Zero W com um módulo Global Positioning System (GPS), um adaptador ELM327 Bluetooth e uma câmera
- Desenvolver um software embarcado no qual será possível verificar a localização atual do veículo, as últimas localizações, capturar imagens e disponibilizar informações da porta OBD
- Desenvolver um aplicativo *mobile* para consultar as informações disponíveis pelo software embarcado
- Notificar o usuário sobre falhas no motor retornadas pela porta OBD

# Fundamentação Teórica

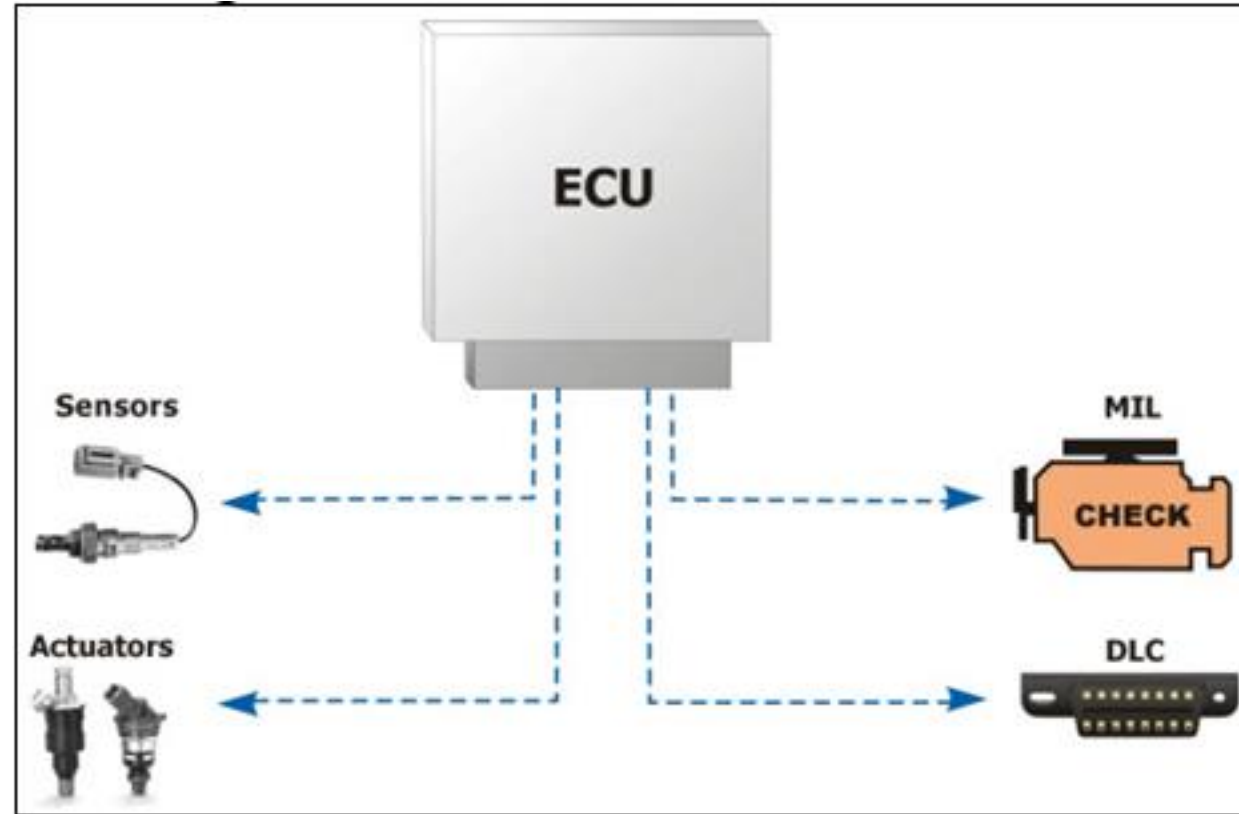
- Internet das Coisas (IoT)
- OnBoard Diagnostic (OBD)
- Raspberry Pi
- Trabalhos Correlatos
- Ferramentas Atuais

# Internet das Coisas (IoT)

- Termo difundido por Kevin Ashton do MIT em 1999
- Controlar objetos remotamente
- Surgiu com a convergência de tecnologias:
  - Redes sem fio
  - Sistemas embarcados
- Principais componentes:
  - As coisas (celulares, sensores, atuadores, computadores)
  - Redes de comunicação

# OnBoard Diagnostic (OBD)

- Disponível na maioria dos veículos
- Obrigatório na Europa e Estados Unidos a partir de 1996
- No Brasil em 2010
- Unidade de Controle Eletrônico (ECU)
- Luz de Mal Funcionamento (MIL)





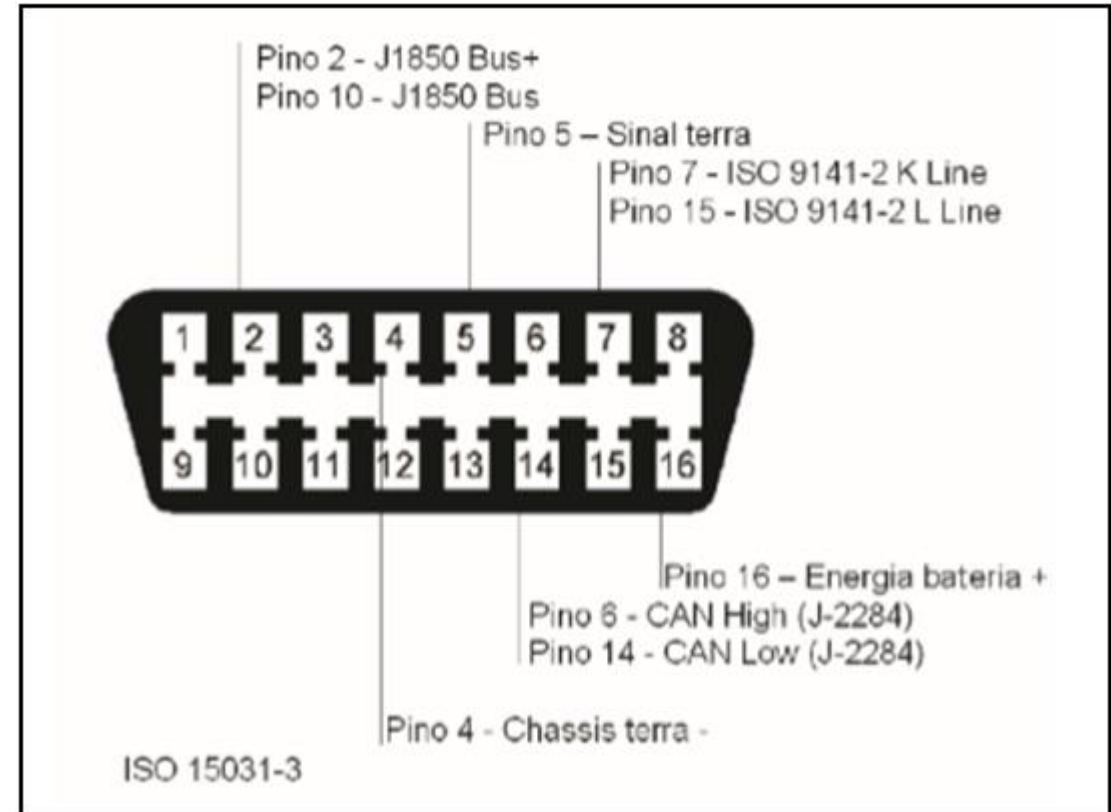
# OBD1

- Falta de padronização entre os veículos
- Itens de diagnóstico
  - Sensor de oxigênio
  - Sistemas de combustível e eletrônico
  - Componentes eletrônicos
  - Códigos de erro



# OBD2

- SAE e ISO criaram normas e padronização de informações entre ECUs e ferramentas
- Conector SAE J1962
- Fácil acesso
- Itens de diagnóstico adicionados:
  - Eficiência e aquecimento de catalisador
  - Sistemas de evaporação e ar secundário
  - Parâmetros do motor
  - Memorização de avarias



# Adaptadores ELM327

- Tipos:
  - RS232: porta serial
  - Universal Serial Bus (USB)
  - Bluetooth
  - Wi-Fi



# Adaptadores ELM327

- Comunicação serial mediante comandos AT
- Comandos internos e comandos para barramento

	ELM327 v1.3a	ELM327 v1.4b	ELM327 v2.2	ELM327L V2.2
Voltagem	4.5V até 5.5V	4.5V até 5.5V	4.2V até 5.5V	2.0V até 5.5V
Modo Low Power (sleep)	Não	Sim	Sim	Sim
configurações Retained on Wake	–	Não	Sim	Sim
RS232 Transmit Buffer Bytes	256	256	512	2048
Comandos AT	93	115	128	128
Verificação de frequência da CAN	Não	Não	Sim	Sim
Suporte de resposta pendente (7F)	Não	Não	Sim	Sim

# Protocolos de Comunicação

Protocolo	Transferência	Observações
SAE J1850 PWM	41.6 Kbps	<ul style="list-style-type: none"><li>• Utilizado pela Ford</li></ul>
SAE J1850 VPW	10.4 Kbps	<ul style="list-style-type: none"><li>• Utilizado pela Chevrolet (GM)</li><li>• Único fio</li><li>• Baixo custo</li></ul>
ISO 1941-2	10.4 Kbps	<ul style="list-style-type: none"><li>• Comunicação assíncrona (UART)</li><li>• Utilizado pela Crysler, fabricantes europeus e asiáticos</li></ul>
ISO 14230	10.4 Kbps	-
ISO 15765	500 Kbps	<ul style="list-style-type: none"><li>• Conhecido como Controller Area Network (CAN)</li><li>• Desenvolvido pela Bosh</li><li>• O padrão ISO 15765-4 determina os requisitos mínimos para uma aplicação OBD</li></ul>

# Serviços de Diagnóstico

- Disponibilizam os dados da ECU
- São organizados por modos de operação e código de Parâmetros (PIDs)
- Serviço 0x00 verifica quais PIDs a ECU do veículo suporta

01	02	03	04	05	06	07	08	09	0A
Sensores	Freeze Frame	DTCs Conf.	Limpar DTCs	Testes Sensores Oxigênio	Monitoram. de componentes	DTCs Pendent.	Alterar Parâm. Motor	Inform. Software Motor	Lista de DTCs

# Diagnostic Trouble Code (DTC)

**P 2 7 2 9**



Números sequenciais

Subsistema ou componente

**0** ou **2**: Controlados pela ISO/SAE;  
**1**: Controlado pelo fabricante;  
**3**: Controlados pelo fabricante e reservados pela ISO/SAE.

**P**: Motor e Transmissão (*Powertrain*);  
**B**: Carroceria (*Body*);  
**C**: Chassis;  
**U**: Rede.

Valor	Descrição
0	Sistema eletrônico completo
1	Controle Ar / Combustível
2	Controle Ar / Combustível; Circuito de injeção
3	Sistema de ignição
4	Controle auxiliar de emissões
5	Controle de velocidade do veículo e rotação em marcha lenta
6	Circuito de entrada e saída da central eletrônica
7	Transmissão
8	Transmissão

# Raspberry Pi

Modelos \ Características	Velocidade	RAM	Portas USB	Ethernet	Wireless e Bluetooth	Preço
Raspberry Pi Model A+	700Mhz	512MB	1	Não	Não	\$20
Raspberry Pi Model B+	700Mhz	512MB	4	Sim	Não	\$25
Raspberry Pi 2 Model B	900Mhz	1GB	4	Sim	Não	\$35
Raspberry Pi 3 Model B	1200Mhz	1GB	4	Sim	Sim	\$35
Raspberry Pi Zero	1000Mhz	512MB	1	Não	Não	\$5
Raspberry Pi Zero W	1000Mhz	512MB	1	Não	Sim	\$10



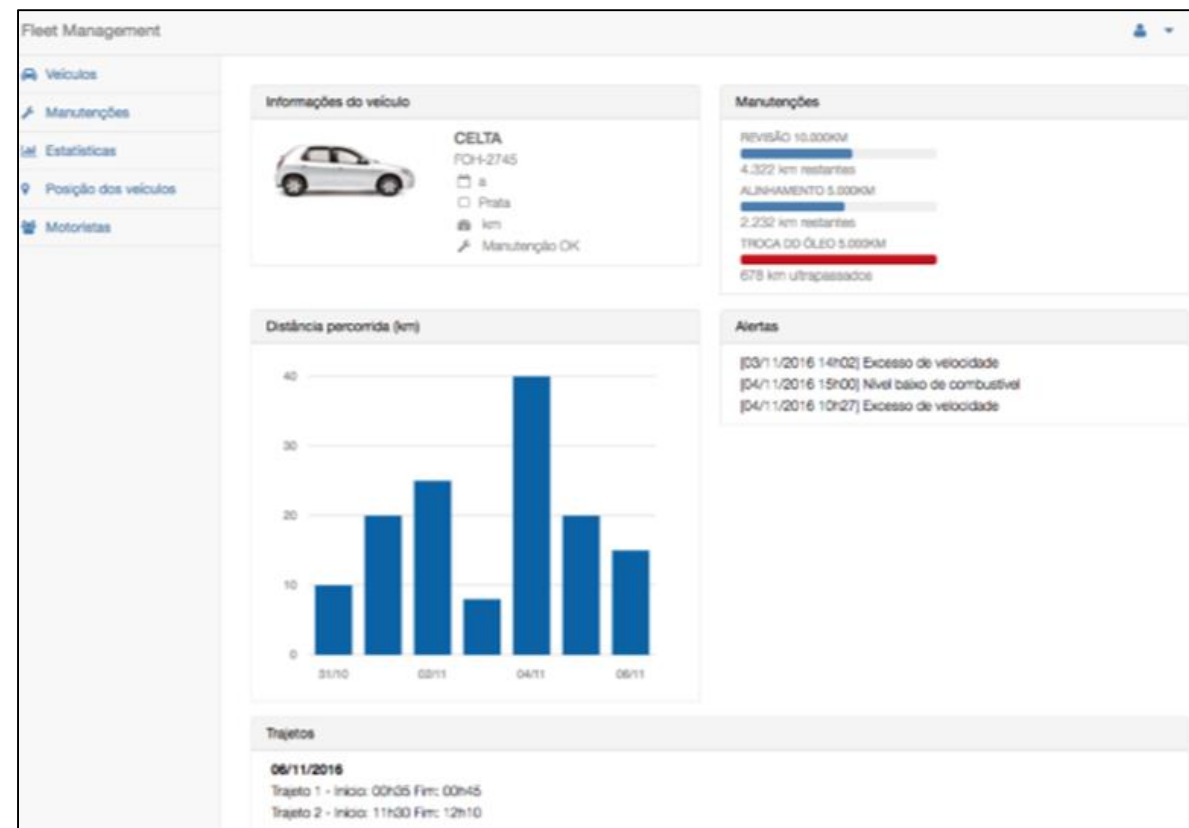


# Trabalhos Correlatos e Ferramentas Atuais

- Gestão de Frota de veículos - Pacheco (2016)
- Localização de veículos para Android - Pina (2015)
- Findcar - Baumgarten (2016)
- OBD-JRP - Staroski (2016)

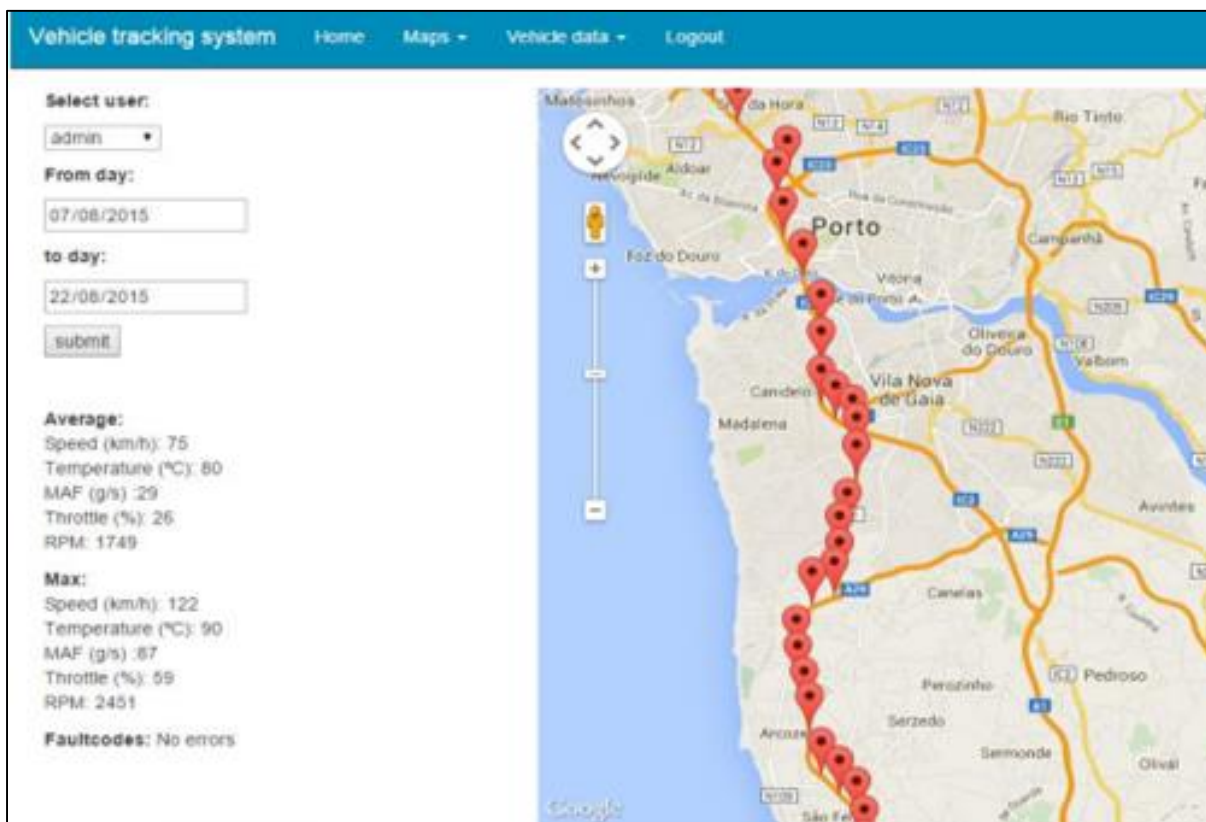
# Gestão de Frota de Veículos – Pacheco (2016)

- Monitoramento de Frota
- Faz a leitura dos sensores do veículo
- Recupera posição geográfica
- Raspberry Pi 3
- Desenvolvido com a linguagem de programação Python e banco de dados SQLite



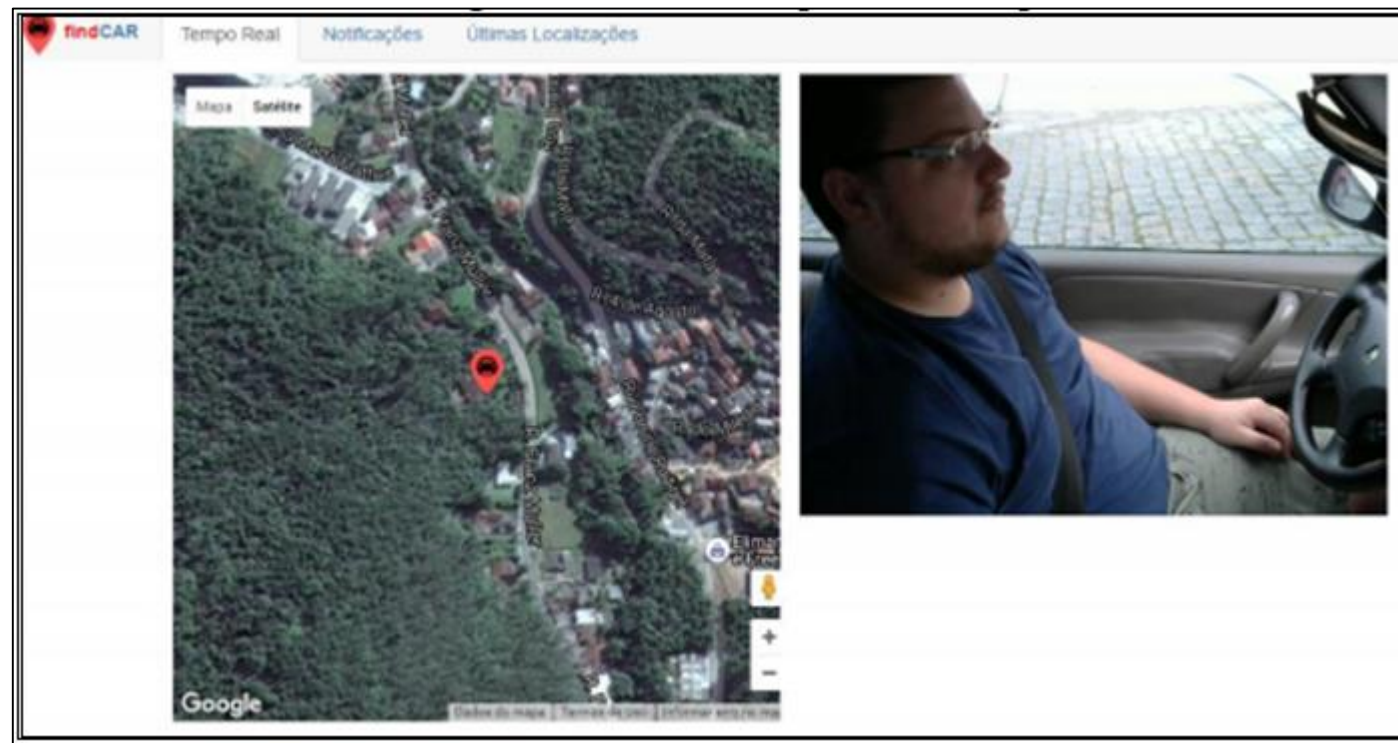
# Localização de veículos - Pina (2015)

- Monitorar localização, sensores OBD2 e também dispositivos Bluetooth por intermédio de uma rede piconet
- Celular Android
- Plataforma web: PHP e MySQL
- Android: JAVA e SQLite



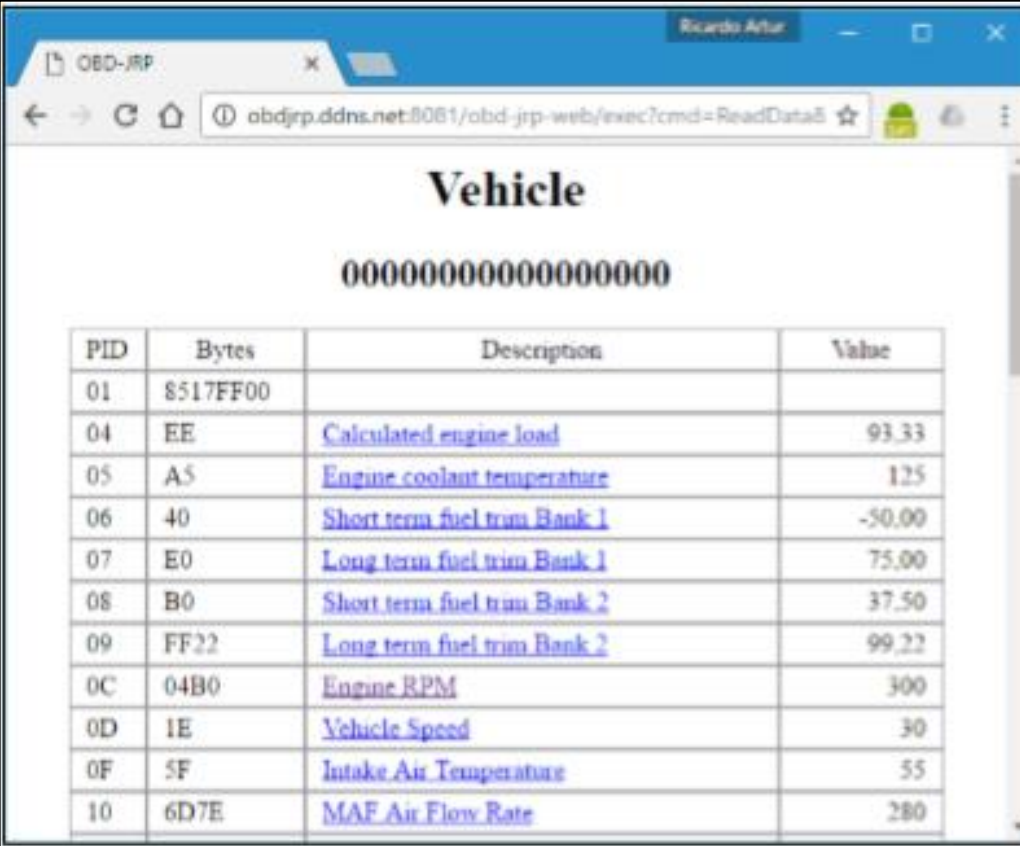
# Findcar - Baumgarten (2016)

- WRTNode + Standard Shield
- OpenWRT
- Monitorar localização e realizar a captura de imagens
- Realizar notificações por e-mail
- PHP Com MySQL



# OBD-JRP - Staroski (2016)

- Capturar dados da porta OBD2
- Raspberry Pi 3
- Firmware desenvolvido com Java utilizando a biblioteca BlueCove
- Web com HTML, CSS e Javascript



The screenshot shows a web browser window with the title 'OBD-JRP'. The address bar contains the URL 'obdjrp.ddns.net:8081/obd-jrp-web/exec?cmd=ReadData5'. The main content area displays the word 'Vehicle' in bold, followed by a long string of zeros '000000000000000000'. Below this is a table with four columns: PID, Bytes, Description, and Value. The table lists various OBD parameters and their current values.

PID	Bytes	Description	Value
01	8517FF00		
04	EE	<a href="#">Calculated engine load</a>	93.33
05	A5	<a href="#">Engine coolant temperature</a>	125
06	40	<a href="#">Short term fuel trim Bank 1</a>	-50.00
07	E0	<a href="#">Long term fuel trim Bank 1</a>	75.00
08	B0	<a href="#">Short term fuel trim Bank 2</a>	37.50
09	FF22	<a href="#">Long term fuel trim Bank 2</a>	99.22
0C	04B0	<a href="#">Engine RPM</a>	300
0D	1E	<a href="#">Vehicle Speed</a>	30
0F	5F	<a href="#">Intake Air Temperature</a>	55
10	6D7E	<a href="#">MAF Air Flow Rate</a>	280

# Especificação

- Requisitos Funcionais e Não Funcionais
- Diagrama de casos de uso
- Diagrama de arquitetura
- Diagrama de atividades para inicialização do servidor
- Diagrama de atividades para sincronização de processos

# Requisitos Funcionais

- Desligar e ligar o sistema operacional
- Reiniciar o sistema embarcado
- Configurar aplicação (e-mail, telefone e notificações)
- Disponibilizar posição geográfica, imagens do veículo e dados da porta OBD2 em tempo real
- Informar estado de execução do servidor embarcado por meio de LEDs
- Notificar o usuário por meio de mensagens de texto e e-mail no caso de falhas DTCs

# Requisitos Não Funcionais

- Hardware:

- Raspberry Pi Zero W
- Adaptador ELM327 Bluetooth
- GPS Ubox GY-GPS6MV2
- Raspberry Pi Camera 1.3
- Modem USB 3G ZTE MF626
- Chip da operadora TIM

- Simuladores:

- OBDSim
- ECU Engine Pro

- Software Embarcado:

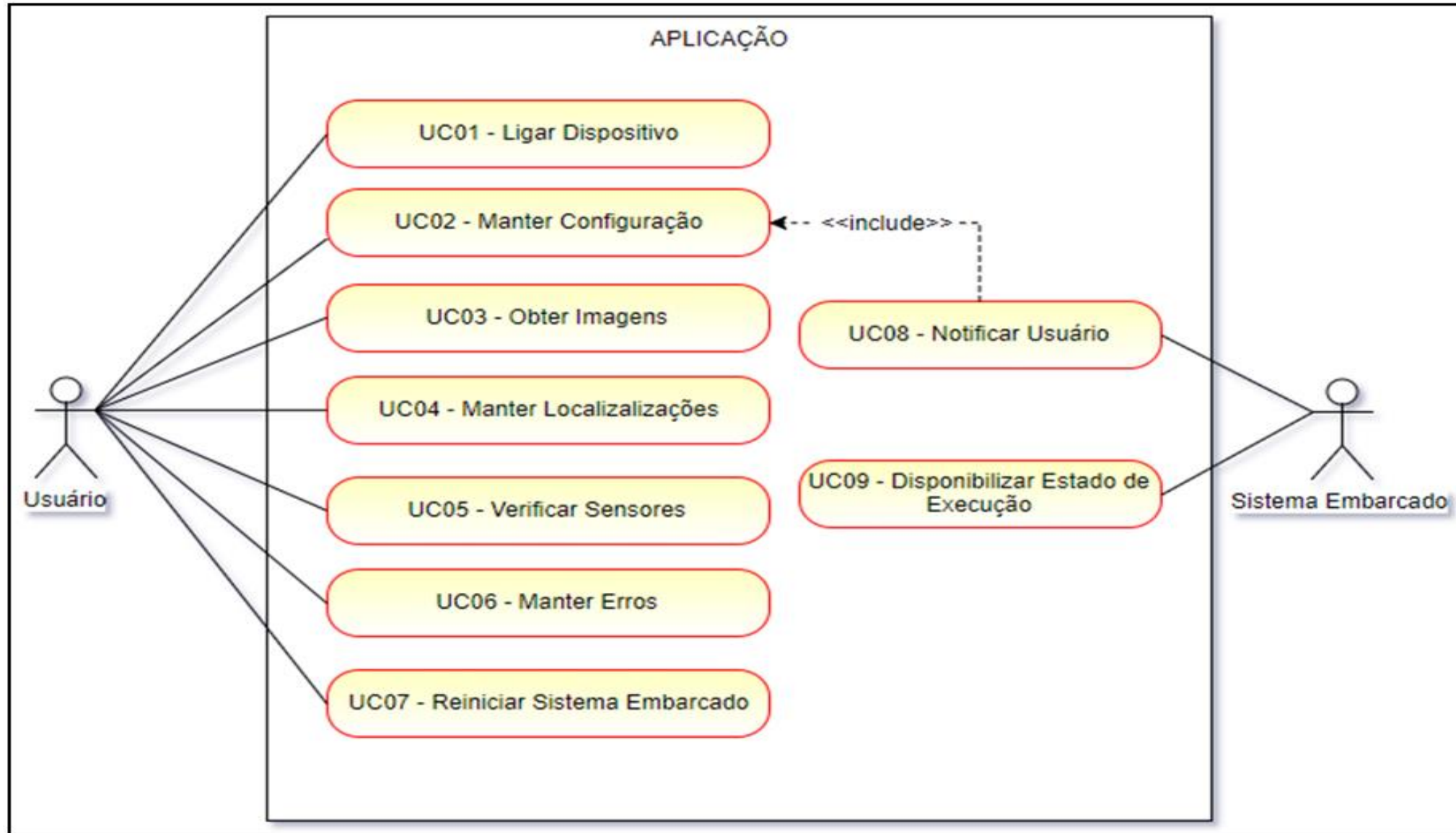
- Sistema operacional Raspian
- Linguagem de programação Python
- Servidor de aplicação Flask

- Aplicativo Mobile:

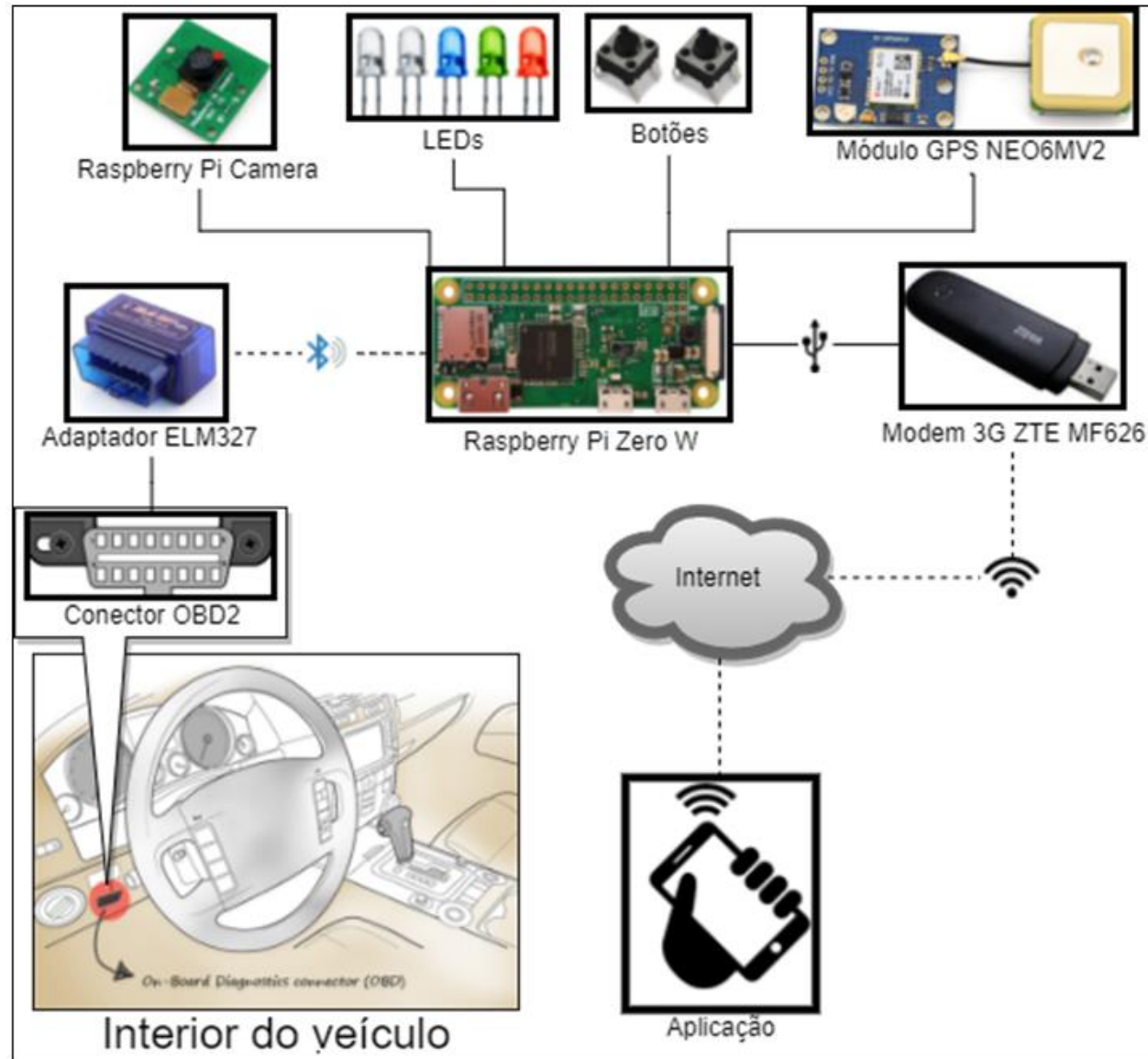
- Linguagem de programação Typescript
- Ionic Framework
- Banco de dados IndexedDB



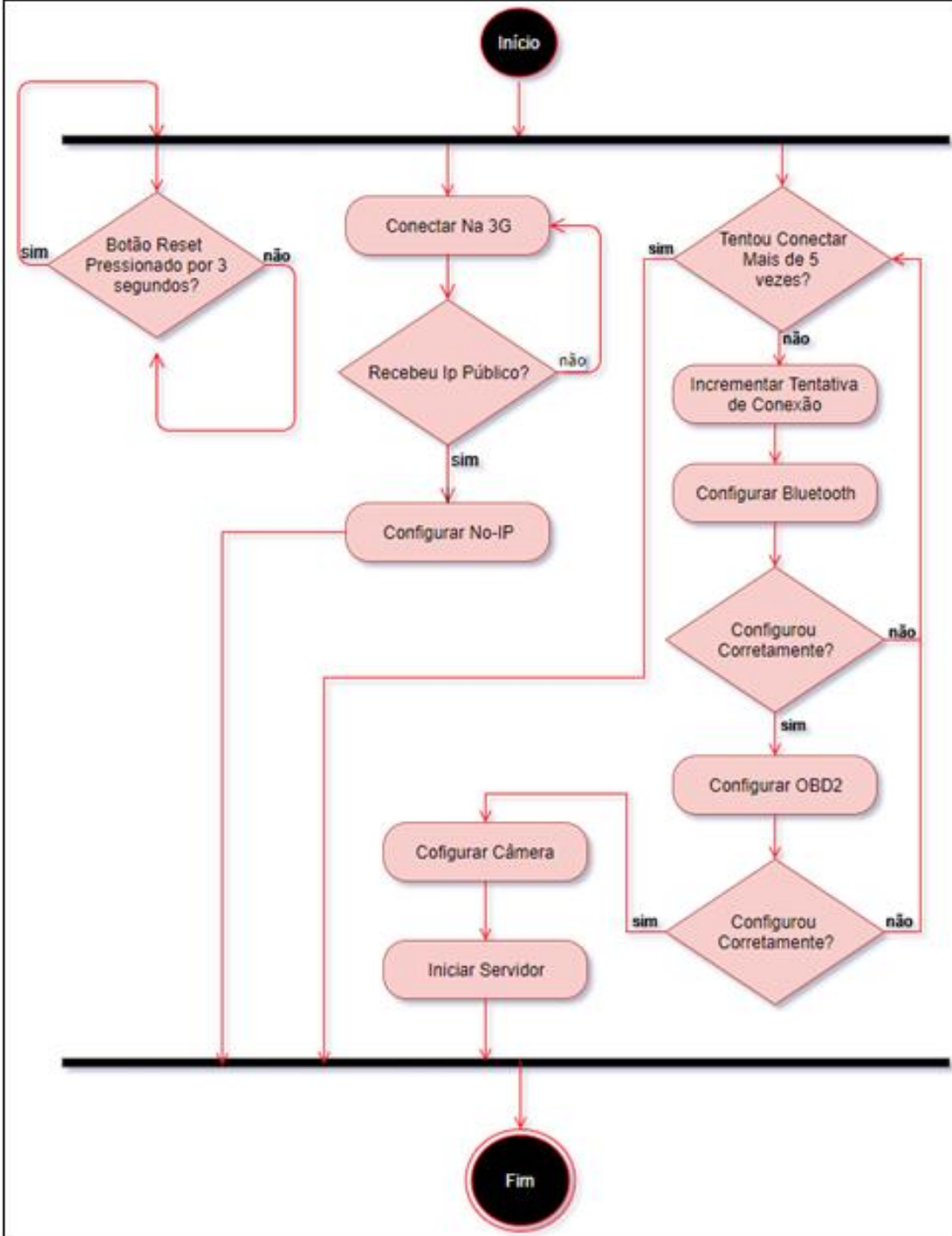
# Diagrama de caso de uso



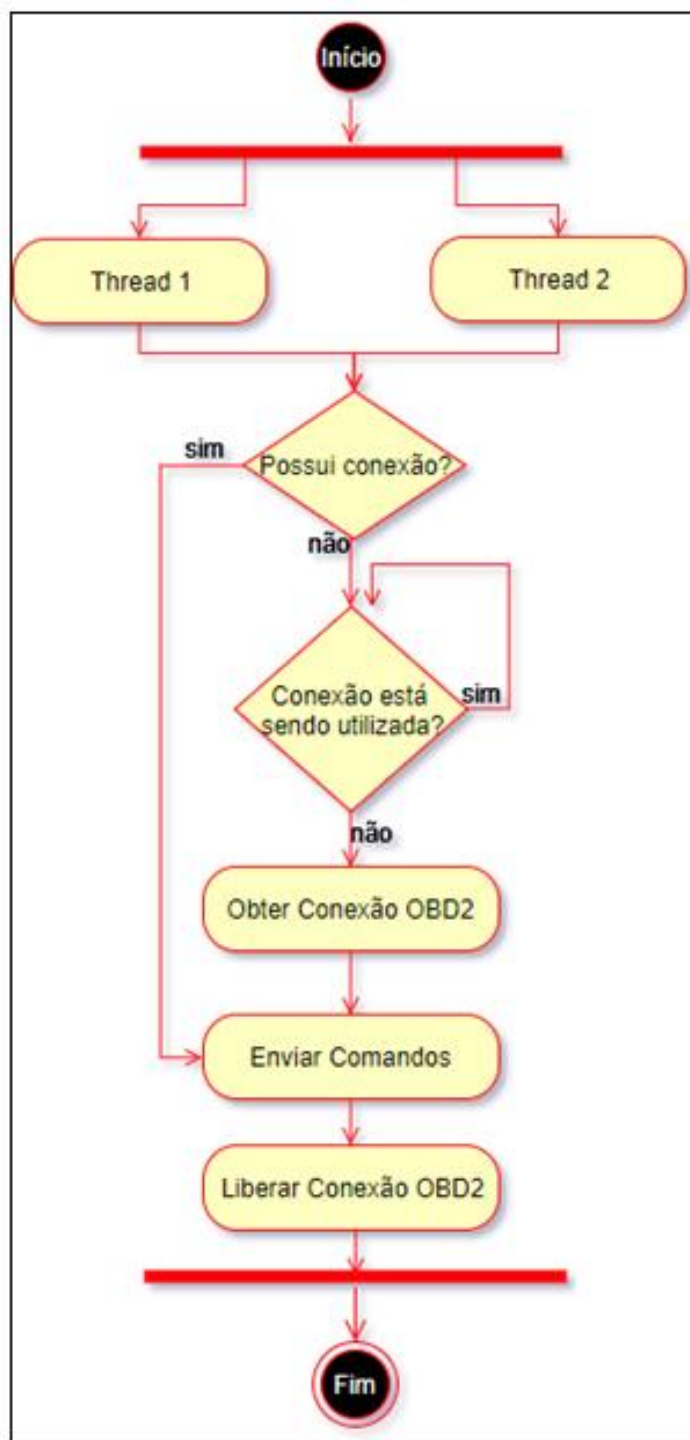
# Diagrama de arquitetura



# Diagrama de atividades para inicialização do servidor



# Diagrama de atividades para sincronização de processos

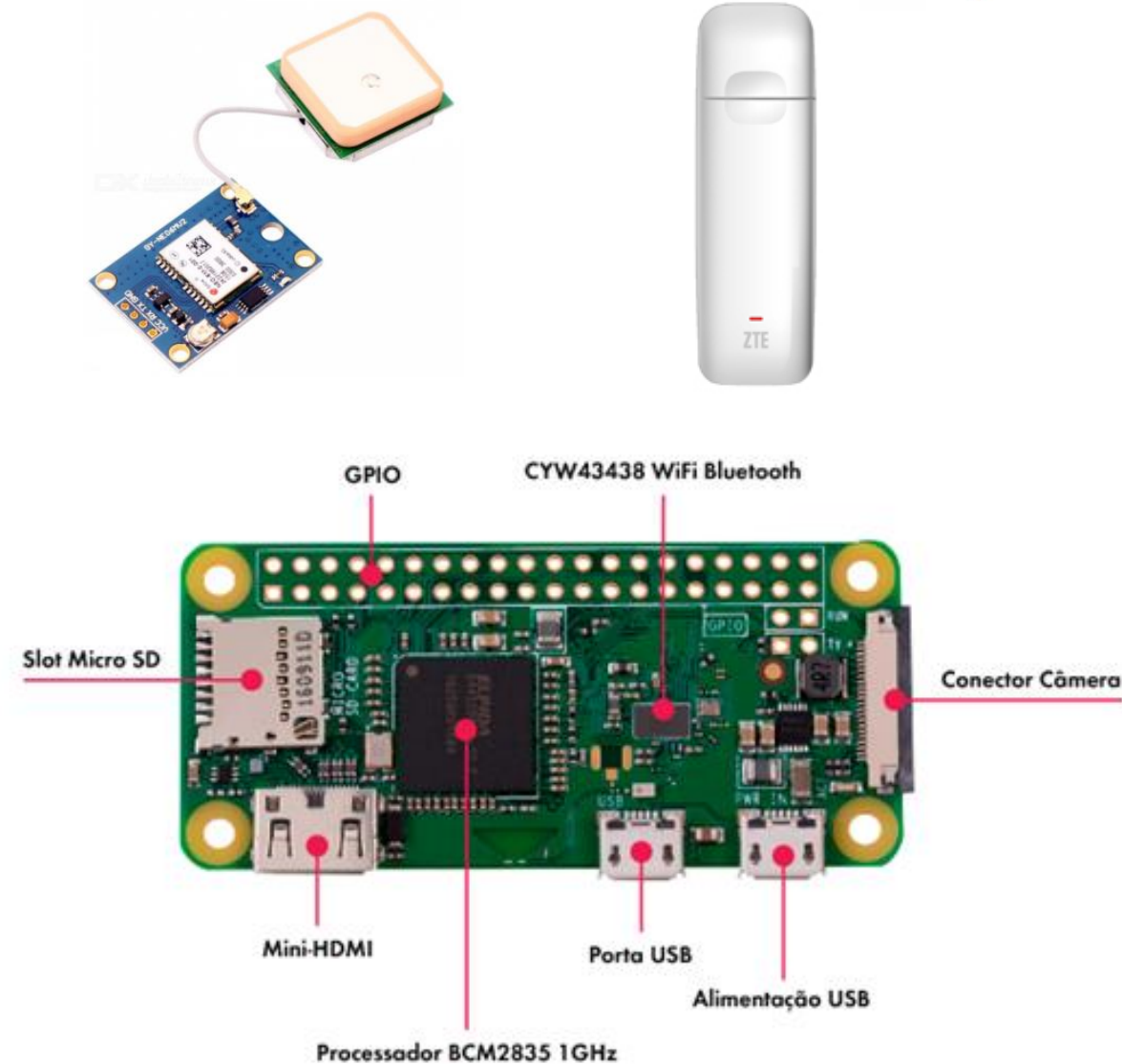


# Implementação

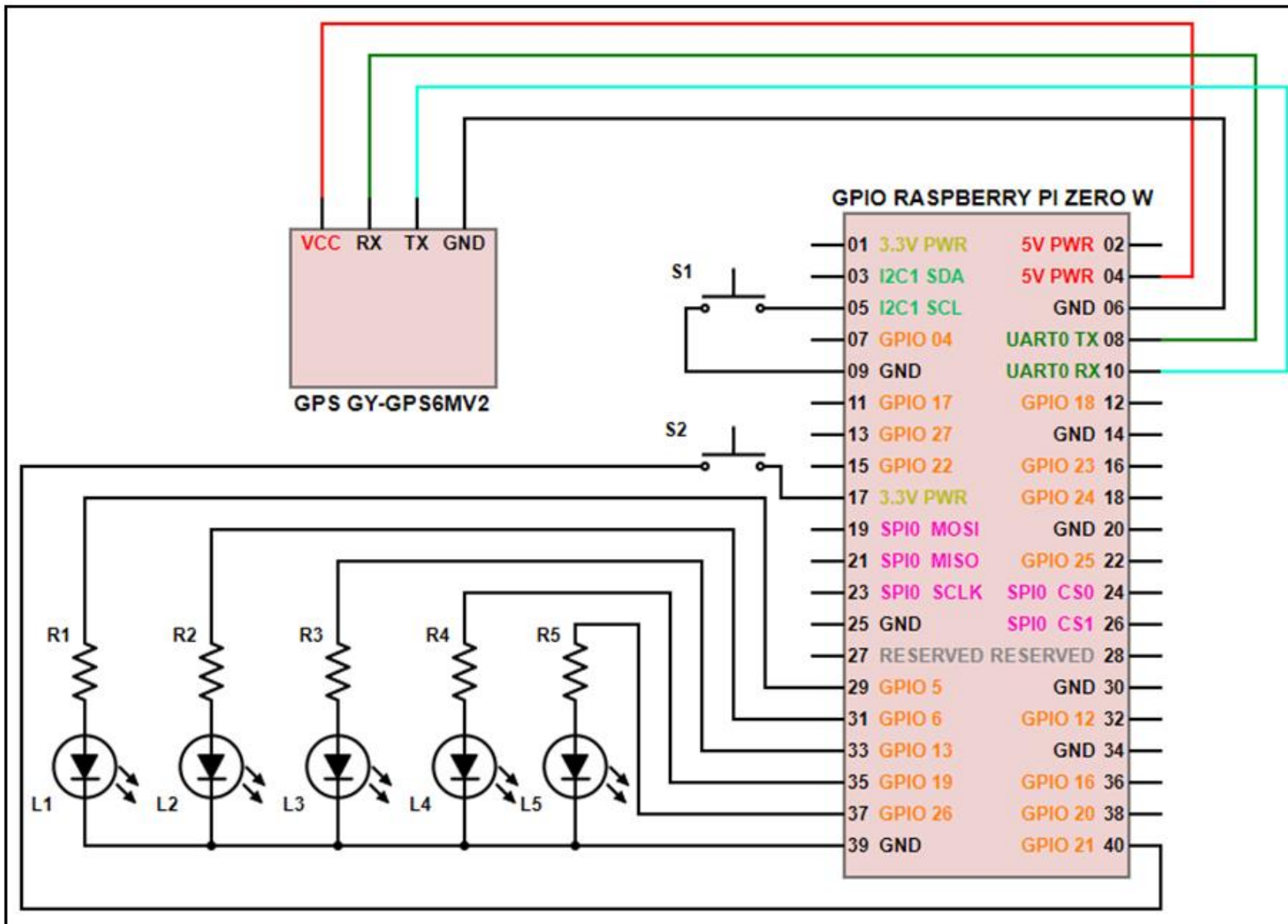
- Construção do Hardware
- Esquema Elétrico
- Hardware Montado
- Técnicas e Ferramentas Utilizadas
- Simuladores De Central Automotiva

# Construção do Hardware

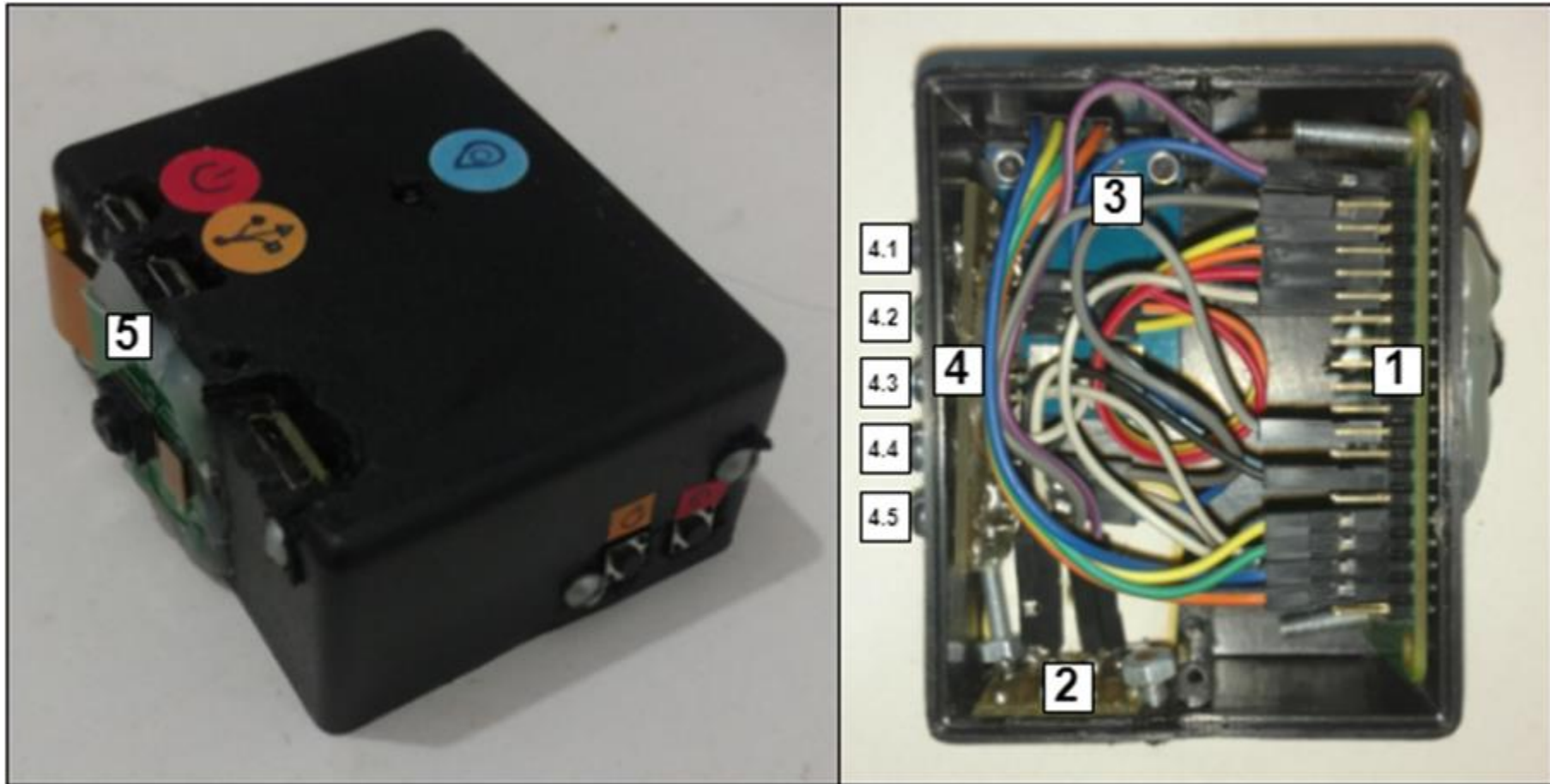
- Raspberry Pi Zero W
- Adaptador ELM327 Bluetooth
- GPS Ubox GY-GPS6MV2
- Raspberry Pi Camera 1.3
- Modem USB 3G ZTE MF626
- LEDs
- Botões Switch



# Esquema Eléctrico



# Hardware Montado





# Técnicas e Ferramentas Utilizadas

- Sistema Embarcado

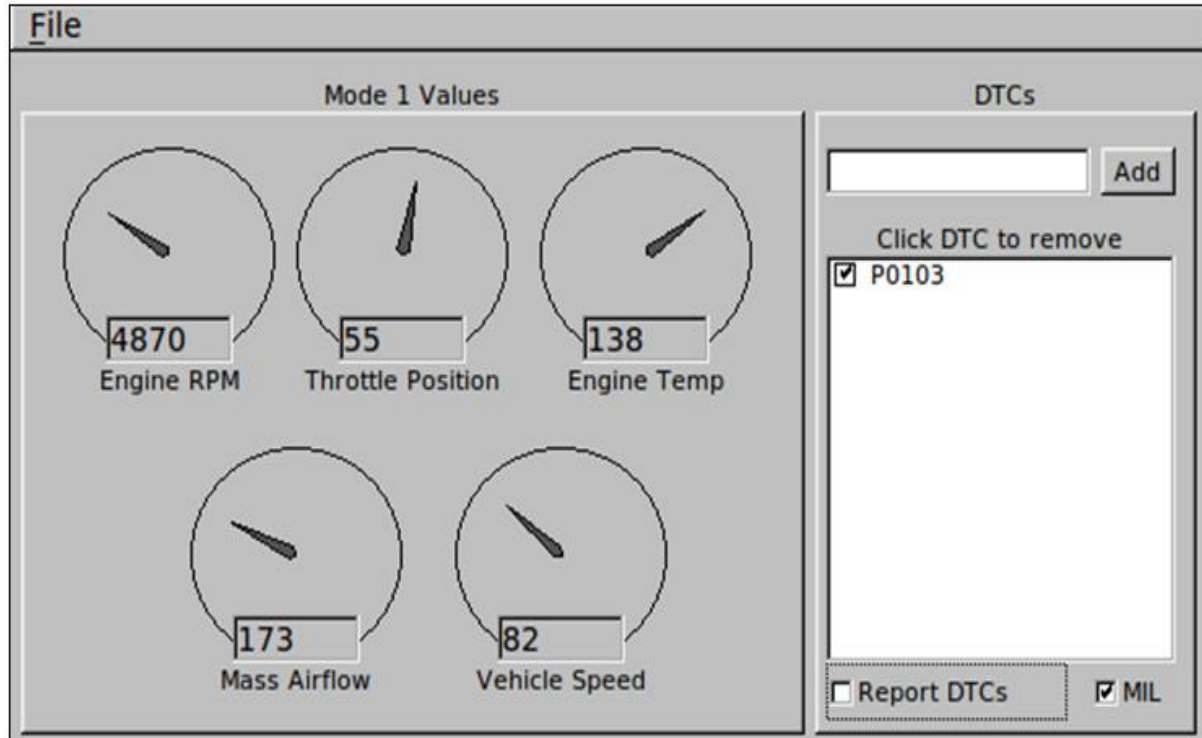
- Python 2.7
- Flask
- Bibliotecas:
  - python-obd
  - flask-video-streaming

- Aplicativo Mobile

- Typescript
- Ionic Framework
- Bibliotecas:
  - AngularJS
  - AgmCoreModule
  - IonicStorage
  - GaugesModule

# Simuladores De Central Automotiva

OBDSim



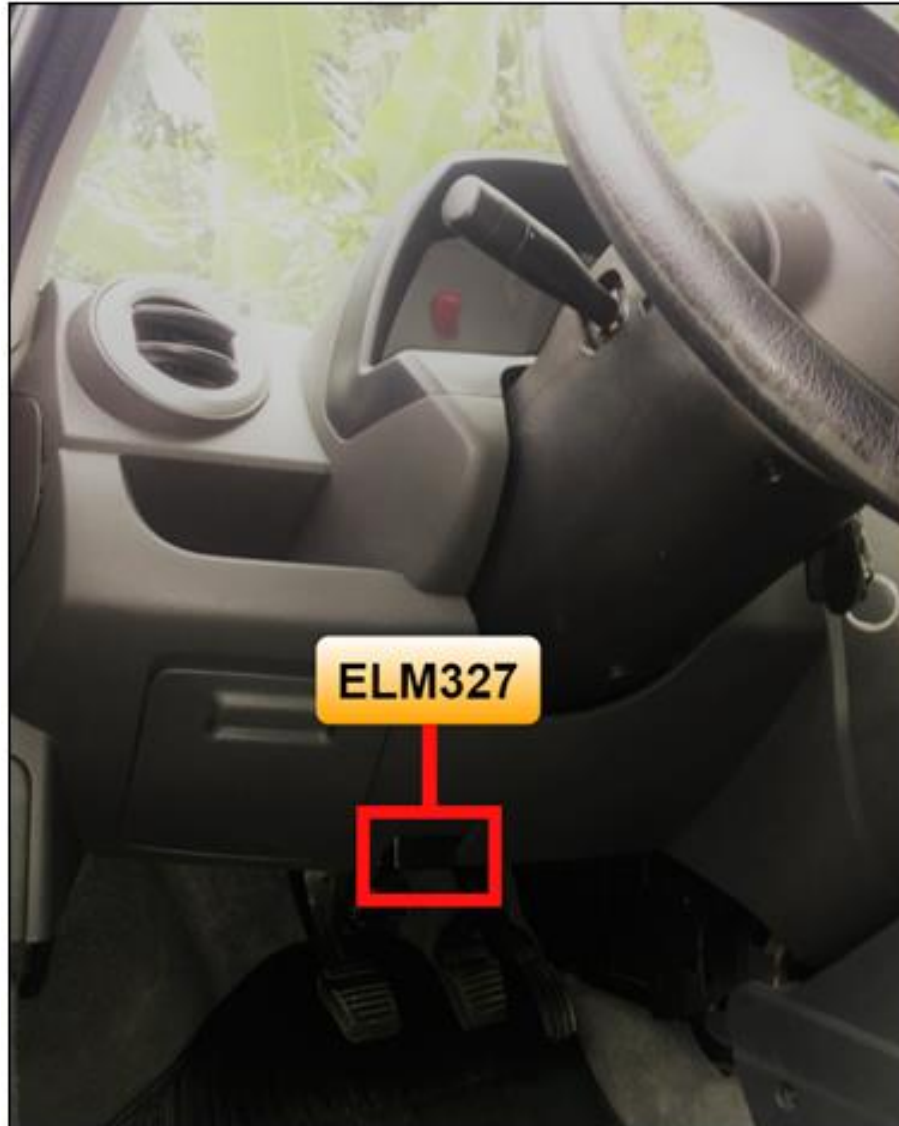
ECU Engine Pro



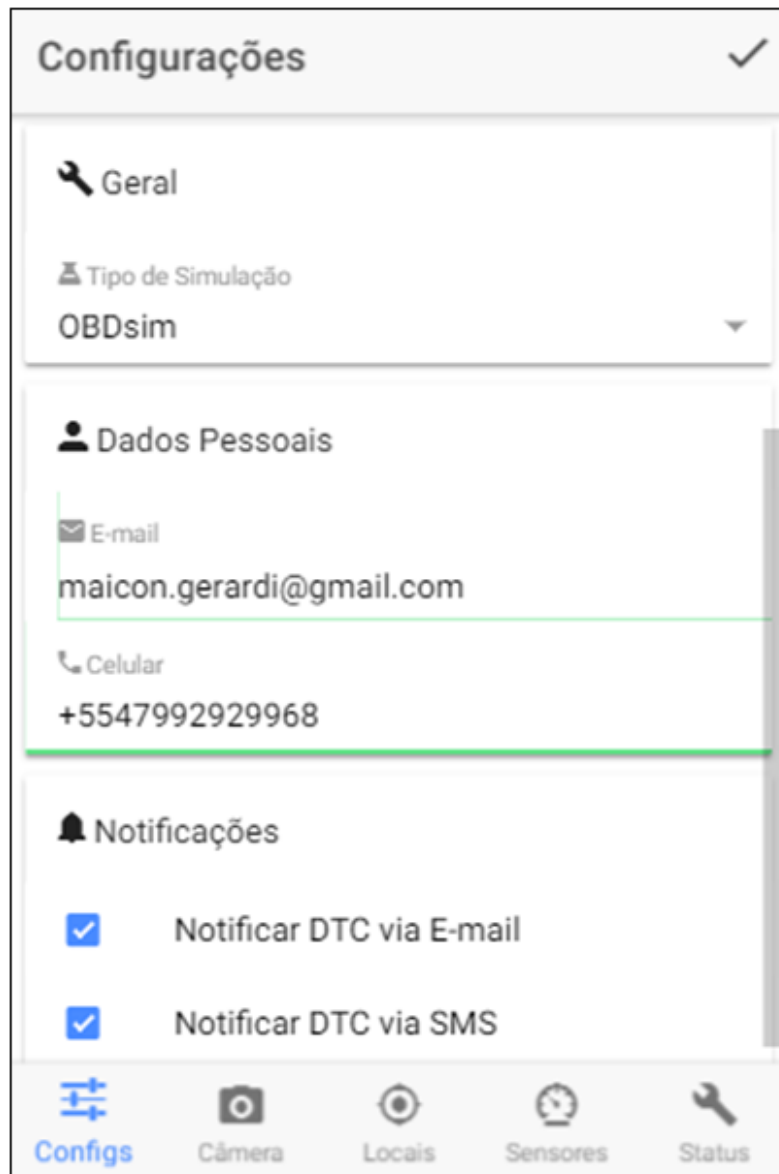
# Operacionalidade da Implementação

- Hardware Instalado
- Aplicativo Mobile

# Hardware Instalado



# Aplicativo Mobile



# Análise dos Resultados

- Modems 3G/4G
- Operadoras De Telefonia Móvel
- Testes Em Veículos
- Experimentos De Falhas DTC
- Comparação Dos Trabalhos

# Modems 3G/4G

- Modem USB 3G/4G Huawei E3272 Hilink
  - Não é possível desabilitar o DHCP
  - Não possui a função Virtual Server
  - Cria uma conexão Ethernet automaticamente não permitindo controle à conexão
  - Acesso ao servidor remoto por meio de um serviço de proxy reverso
- Modem USB 3G ZTE MF626
  - Cria uma conexão direta com a operadora por meio de PPP

Conexão	Leitura dos Sensores (s) *
<i>Wireless</i>	1,112
3G	1,22
3G com <i>proxy</i>	1,53

\* Dados baseados em uma leitura completa dos sensores do simulador OBDSim

# Operadoras de Telefonia Móvel

<b>OI</b>	<b>VIVO</b>	<b>CLARO</b>	<b>TIM</b>
Não foi testada por problemas no chip adquirido	Não obteve IP público	Não obteve IP público	Obteve IP público

- Planos Pré-pago
- Testes de velocidade:
  - Speedtest:
    - Ping: 89 ms
    - Download: 2,96 Mbps
    - Upload: 0,49 Mbps
  - Fast.net
    - Download: 2,3 Mbps



# Testes Em Veículos

<b>Veículo</b>	<b>Qtde PIDs</b>	<b>Protocolo</b>	<b>Tempo de Leitura (s)*</b>
Volkswagen Fox City 2005/06	25	ISO 9141-2	6
Ford Ecosport 2006/07	7	ISO 15765-4 (CAN 11/500)	1
Volkswagen Gol 2009/10	23	ISO 14230-4 (KWP FAST)	2
Ford Ka 2011/12	26	ISO 15765-4 (CAN 11/500)	2
Peugeot 306 2012/13	8	ISO 15765-4 (CAN 11/500)	1
Ford Edge 2013	77	ISO 15765-4 (CAN 11/500)	14
Volkswagen Fox Run 2016/17	50	ISO 15765-4 (CAN 11/500)	9

\* Dados baseados em uma leitura completa dos sensores

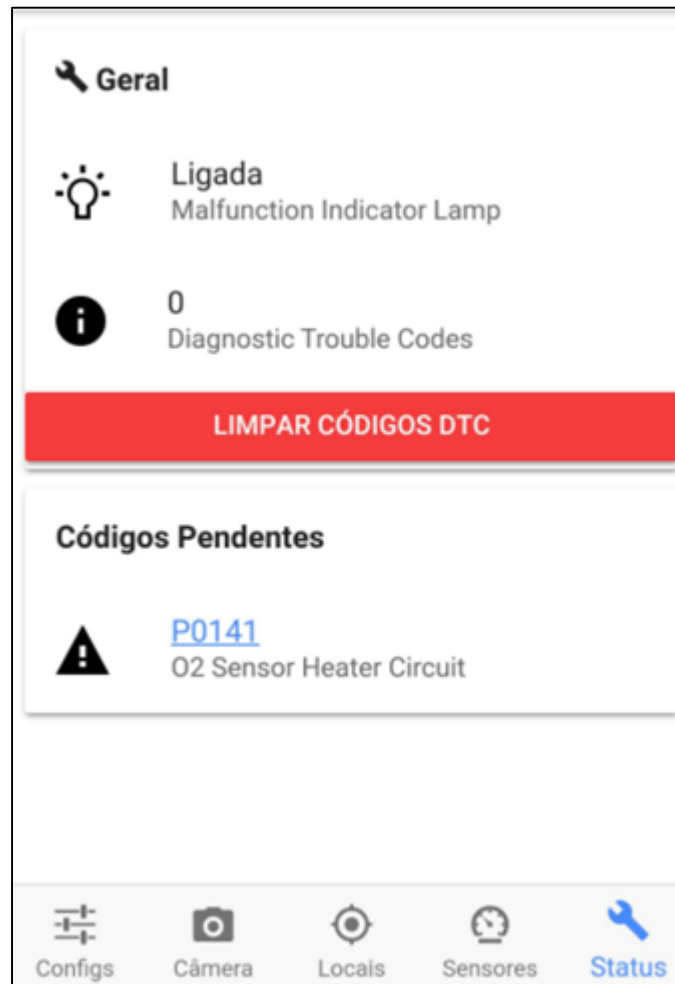
# Experimentos De Falhas DTC

- Recuperar códigos DTC

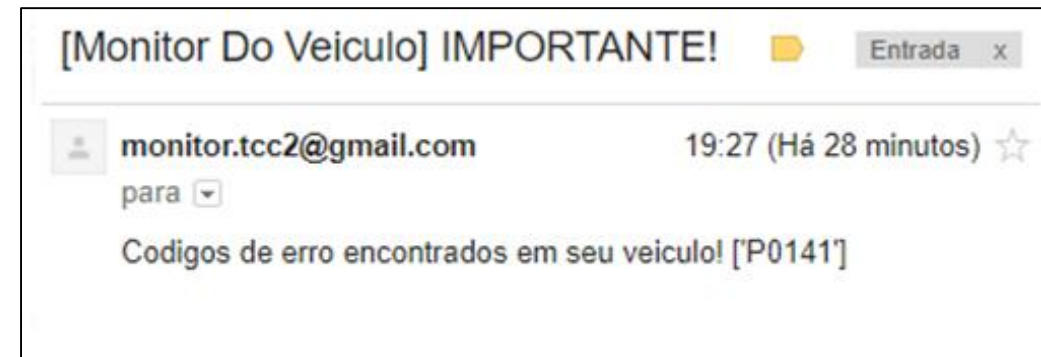
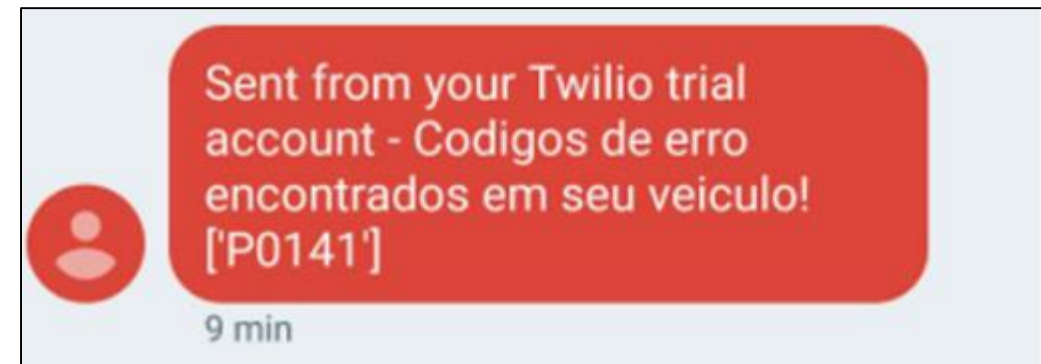


# Experimentos De Falhas DTC

- Tela do Aplicativo



- Notificações Recebidas



# Comparação Dos Trabalhos

Trabalhos Características	Pacheco (2016)	Pina (2015)	Baumgarten (2016)	Staroski (2016)	Aplicação Desenvolvida
Hardware principal do sistema embarcado	Raspberry Pi 3	Celular Android	WRTnode	Raspberry Pi 3	Raspberry Pi Zero W
Custo do Hardware	U\$ 35	R\$ 567	R\$ 117	U\$ 35	U\$ 10
Consumo (ampere)	0,800	1,8	0,300	0,800	0,180
Leitura de código de erro (DTC)	Não	Não	Não	Não	Sim
Monitoramento de uma frota	Sim	Não	Não	Não	Não
Leitura dos sensores do veículo	Sim	Sim	Não	Sim	Sim
Recupera dados de posição geográfica	Sim	Sim	Sim	Não	Sim
Utilização em dispositivos móveis	Não	Sim	Não	Não	Sim
Captura de imagens em tempo real	Não	Não	Sim	Não	Sim

# Conclusões e Sugestões

# Conclusões

- Biblioteca python-obd facilitou a implementação dos sensores e DTCs
- Placa Raspberry Pi Zero W disponibiliza:
  - Recursos necessários para o projeto
  - Economicamente acessível
  - Baixo consumo energético
- Importante por viabilizar uma forma de acesso ao veículo através da Internet
- Objetivos do trabalho foram alcançados

# Sugestões

- Utilizar a distância entre o veículo e smartphone para notificar possíveis furtos
- Alterar a potência do motor por meio do modo de diagnóstico 0x08
- Cortar o combustível no caso de furto
- Utilizar o módulo Pi Noir Camera para imagens noturnas
- Fazer uso de Inteligência Artificial para detectar manutenções e avarias
- Expandir a aplicação para atender veículos pesados como caminhões e ônibus pelo padrão Fleet Management System (FMS)

Demonstração