Universidade Federal de Santa Catarina
Centro de Blumenau
Departamento de Engenharia de
Controle, Automação e Computação

André Menezes Melo e Silva

Sistema para rastramento veicular via GPS

Blumenau
2021

André Menezes Melo e Silva

Sistema para rastramento veicular via GPS

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Santa Catarina como parte dos requisitos necessários para a obtenção do Título de Engenheiro de Controle e Automação.
Orientador: Prof. Dr. Carlos Moratelli

Universidade Federal de Santa Catarina
Centro de Blumenau
Departamento de Engenharia de
Controle e Automação e Computação

Blumenau
2021

André Menezes Melo e Silva

Sistema para rastramento veicular via GPS

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de Engenheiro de Controle e Automação.

Comissão Examinadora

Documento assinado digitalmente
Carlos Roberto Moratelli
Data: 23/05/2021 17:05:07-0300
CPF: 035.999.369-94
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Carlos Moratelli
Universidade Federal de Santa Catarina
Orientador

Documento assinado digitalmente
Maiquel de Brito
Data: 24/05/2021 06:40:39-0300
CPF: 979.657.150-15
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Maiquel de Brito
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Marcos Vinicius Matsuo
Data: 24/05/2021 07:43:29-0300
CPF: 052.580.029-85
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Marcos Matsuo
Universidade Federal de Santa Catarina

Blumenau, 21 de maio de 2021

Dedico este trabalho a todos aqueles que, de alguma forma,
auxiliaram para a concretização desta etapa.

Agradecimentos

Ao professor e orientador Carlos Moratelli, e aos demais professores que de alguma forma participaram na minha formação.

Ao meu pai e minha mãe que nunca deixaram de prestar o suporte necessário para que eu pudesse superar todas as etapas deste curso e chegar à condição de formando.

Finalmente aos meus amigos, que sempre estiveram prontos para apoiar direta ou indiretamente durante todo o curso, como também na realização deste trabalho.

"Train yourself to let go of everything you fear to lose."
(Yoda)

Resumo

Dentro da área de informação e tecnologia o ramo de segurança vem sendo uma parte indispensável e vem crescendo cada vez mais conforme os índices de criminalidade aumentam. Existem diversas empresas de tecnologia voltadas apenas para a segurança, dentre os diferentes setores que este ramo possui, o de segurança veicular é um deles e será tratado neste trabalho. Visto isso, este trabalho trata do desenvolvimento de um sistema voltado para a segurança pessoal. O mesmo faz a integração de três diferentes setores, o hardware que é um dispositivo responsável por coletar as coordenadas geográficas do automóvel através de um GPS(*Global Positioning System*), um serviço web para armazenar e pré visualizar esses dados e por fim um aplicativo para sistemas Android que recebe essas coordenadas através de um módulo GSM(*Groupe Special Mobile*) e tem a responsabilidade de apresentar de forma interativa a localização atual do veículo em tempo real. Com o desenvolvimento e a integração de cada um dos sistemas propostos, foi obtido um produto final que traz de forma didática a ideia inicialmente de apresentar ao usuário através de um aplicativo a localização atual do seu automóvel.

Palavras-Chave: 1. GPS. 2. GSM. 3. Sistemas embarcados. 4. IoT. 5. Segurança.

Abstract

Security has been an indispensable branch within the information and technology area, growing more and more as crime rates increase. There are several technology companies focused only on security, among the different ramifications this sector has, there is vehicular security, which is going to be regarded in this project. In view of this, this work covers the development of a system focused on personal security. It integrates three different sectors: the hardware, which is a device responsible for collecting geographic coordinates of the car through a GPS (*Global Positioning System*), a web service for storing and pre-visualizing the data and and application for Android systems that receives these coordinates through a GSM (*Groupe Special Mobile*) module and is responsible for interactively presenting the vehicle's current location in real time. A final product that elucidates didactically the initial idea of showing the user his car's current location through an app, with the development and integration of each of the aforementioned systems, was achieved.

Keywords: 1. GPS. 2. GSM. 3. Embedded system. 4. IoT. 5. Security.

Lista de figuras

Lista de tabelas

Lista de Siglas e Abreviaturas

UFSC	*Universidade Federal de Santa Catarina*
IOT	*Internet Of Things*
GPS	*Global Positioning System*
GPRS	*General Packet Radio Service*
GSM	*Global System for Mobile*
ROM	*Read-Only Memory*
RAM	*Random Access Memory*
CPU	*Central Processing Unit*
USB	*Universal Serial Bus*
GND	*graduated neutral density filter*
ICSP	*In Circuit Serial Programming*
LED	*Light Emitting Diode*
SPS	*Standard Positioning Service*
PPS	*Precise Positioning Service*
RTCP	*Real-Time Transport Control Protocol*
IP	*Internet Protocol*
SIM	*Subscriber Identity Module*
MBPS	*Megabits por segundo*
JSON	*JavaScript Object Notation*
XML	*Extensible Markup Language*
MQTT	*Message Queue Telemetry Transport*
SOC	*System On Chip*
API	*Application Programming Interfacep*

Sumário

1 Introdução

1.1 Contextualização do problema

Há algum tempo a criminalidade ocupa uma grande parte dos noticiários. Este é um problema que abrange grande parte das nações, entre elas uma das nações mais atingidas é o Brasil. Muitos fatores influenciam diretamente no crescimento da taxa de criminalidade, como a miséria, a pobreza, a desigualdade social, a cultura das más companhias, entre outros [1].

Uma das ações criminais que mais vem acontecendo nos últimos anos é o roubo e furto de automóveis. De acordo com o Sinesp (Sistema Nacional de Informações de Segurança Pública) foram registradas 1.103.606 ocorrências de roubo e 1.139.961 de ocorrências de furto de veículos no Brasil, totalizando mais de 2 milhões de ocorrências de roubo ou furto entre os anos de 2014 e 2019 o que representa uma média de mais de 100 carros roubados para cada 100 mil habitantes [2].

A quantidade de veículos furtados e roubados no brasil vem avançando de forma significativa, o número de roubos aumenta proporcionalmente conforme o aumento no número de automóveis em circulação, a principal utilidade desses automóveis roubados é para a venda ilegal em outros estados ou para pratica de crimes, como sequestros ou transporte de drogas.

Os principais problemas relacionados ao roubo de automóveis é o risco de vida que os donos do veículo são expostos, principalmente devido a tentativas de reação ao roubo, que ocorrem principalmente pela incerteza de recuperação do bem pelas autoridades, além da possibilidade de sequestro que os donos do veículo são expostos.

Visto este cenário, é necessário uma solução que proporcione de forma prática, rápida e segura a localização do veículo em tempo real que implicaria diretamente nas chances de recuperação do veículo, além de tornar mais difícil a utilização dos veículos furtados para outras práticas criminosas.

O sistema desenvolvido possibilita ao usuário fazer o rastreamento do seu veículo em tempo real através de coordenadas geográficas obtidas por um GPS, sendo necessário para utilizar esse serviço ter apenas o sistema instalado em seu veículo e um aparelho móvel Android com o aplicativo. No aplicativo será fornecido a localização do automóvel bem como a rota entre o usuário e o veículo, esse sistema tem a função de melhorar a segurança do veículo do usuário contra furtos e roubos e aumentar as chances de recuperação do bem.

1.2 Objetivos Gerais

O objetivo deste projeto de forma geral é o desenvolvimento de um sistema para rastreamento veicular via GPS de baixo custo baseado na plataforma Arduino, que possa auxiliar os usuários no rastreamento dos veículos diminuindo as chances de furto ou roubo, minimizando danos aos proprietários e outros passageiros envolvidos na ação criminosa e aumentando a chance de recuperação do veículo.

1.3 Objetivos especificos

Dentre os objetivos específicos citam-se os seguintes itens:

- Desenvolvimento do *hardware* e *software* para obter os dados de localização do veículo em tempo real, através da placa controladora Arduino Uno e do módulo SIM808, que é composto por um módulo de GPS, para obtenção das coordenadas de localização do veículo, e por um módulo de GSM/GPRS utilizado para enviar os dados obtidos.
- Integração do *hardware* desenvolvido com um *webservice* (serviço de servidor na web) para armazenamento e visualização dos dados obtidos pelo GPS.
- Desenvolvimento de um aplicativo para *Android* para que de forma mais simples e interativa seja possível observar a localização atual do veículo, bem como uma rota entre o usuário do aplicativo e o veículo no qual o sistema está instalado.
- Integração dos três sistemas citados de forma que seja possível a obtenção e envio dos dados para a parte final (aplicativo) em tempo real, para isso sendo necessário então a obtenção dos dados de localização geográfica pelo GPS, o recebimento desses dados pelo *webservice* e então o envio para a parte final do sistema, o aplicativo.

2 Referencial tecnológico

Este capítulo tem por objetivo expor uma revisão teórica sobre os principais temas e tecnologias referentes ao desenvolvimento deste trabalho. Serão feitas revisões teóricas sobre sistemas embarcados, microcontrolador, Arduino, GPS, comunicação móvel, Web-services, entre outros.

2.1 Sistemas embarcados

Um sistema embarcado é qualquer dispositivo que inclui um computador programável, que é utilizado para resolver ou amenizar problemas de cunho específico. Um sistema embarcado opera geralmente com poucos recursos e executa algoritmos buscando resolver problemas e obter respostas em tempo real [3].

Segundo LI et. al [4]: "O adjetivo embarcado reflete o fato desses sistemas serem usualmente parte integrante de um sistema maior. No entanto, apesar de muitos sistemas embarcados poderem coexistir em um sistema, eles podem, por si só, representar o sistema completo e operar individualmente."

Nos dias atuais, os sistemas embarcados estão, de alguma forma, presentes em todos os segmentos do dia-a-dia. Eles podem ser encontrados em diferentes áreas como indústria automotiva, aeronáutica, agronomia e área médica [5]. Estes sistemas, geralmente são caracterizados por apresentarem maior confiabilidade, comparado a outros sistemas computacionais [6].

Com a revolução tecnológica e a ascensão da internet das coisas (IoT) os sistemas embarcados estão se tornando cada vez mais populares e mais presentes no dia-a-dia como em celulares, semáforos, aparelhos de ar condicionado, impressoras entre outros. A internet das coisas de forma geral é uma rede de objetos físicos que possuem tecnologia embarcada e a capacidade de, através de conexão com rede, coletar e transmitir dados, possibilitando uma grande comunicação entre vários objetos e usuários, como se fosse um sistema nervoso que permite a troca de informações entre dois ou mais pontos.

O que torna os sistemas embarcados tão populares, além de sua fácil integração são algumas características, como, o seu tamanho e falta de complexidade em sua composição, que além de ocupar menos espaço e facilitar em sua locomoção torna o custo do sistema embarcado reduzido uma vez que não é necessário tanto poder computacional para executar suas tarefas. Sistemas embarcados estão se tornando cada vez mais utilizados por serem computadores dedicados e então serem utilizados para realizar tarefas restritas o que torna mais fácil e prática a programação desses sistemas.

2.2 Microcontrolador

Um microcontrolador nada mais é do que um circuito integrado que é composto por uma série de componentes que juntos fornecem o necessário para o seu funcionamento, dependendo apenas de uma fonte de alimentação externa. Pode-se definir um microcontrolador como um computador de um único chip. A Figura 1 ilustra os componentes característicos de um microcontrolador [7].

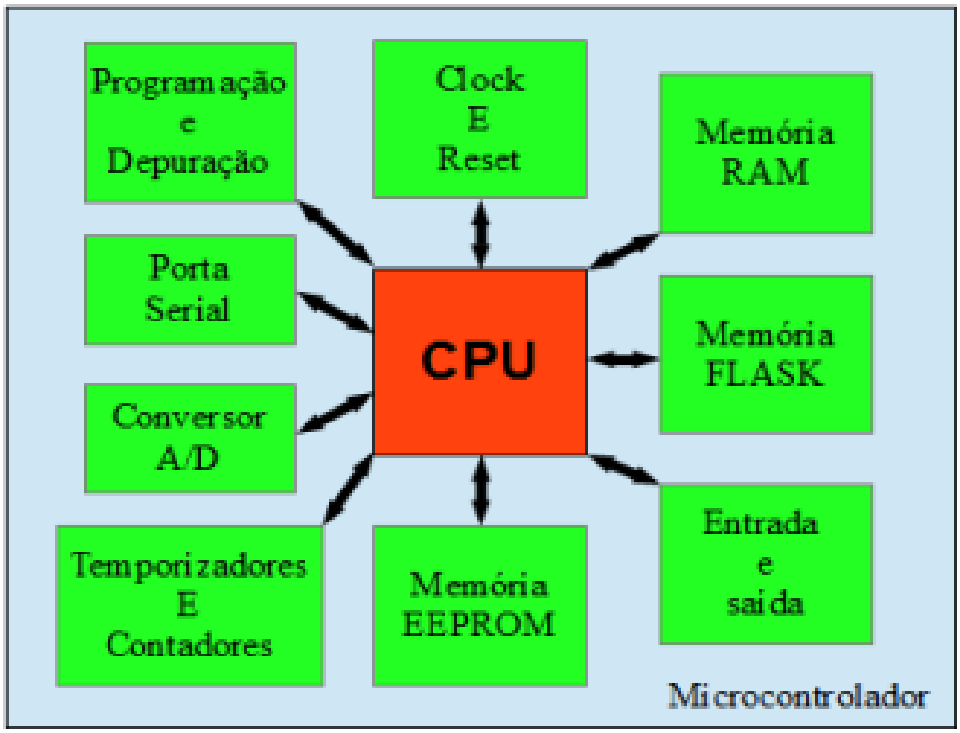


Figura 1 – Componentes de um microcontrolador [7].

Os microcontroladores reúnem em um único componente os elementos de um sistema microprocessado completo (memória ROM, memória RAM, interface paralela, interface serial, temporizador, contador de eventos, controlador de interrupções, entre outros). A parte mais importante do microcontrolador é o microprocessador ou a Unidade de Processamento Central (CPU, Central Processing Unit). A principal diferença de um microcontrolador e um sistema tradicional é que o microcontrolador é um *System On Chip* (SOC), ou sistema-em-um-chip, ou seja apresentam os seus componentes integrados em um circuito, o que o torna compacto, mais barato e de mais fácil uso, além de apresentar um menor consumo de energia.

Este componente têm possibilitado o desenvolvimento de equipamentos que atendam as necessidades de projetos e que sejam facilmente manipuláveis para atender cada necessidade, então os microcontroladores estão cada vez mais sendo utilizados em aplicações que envolvam produtos e dispositivos automatizados, como em máquinas industriais, sistemas de supervisão, sistemas de controle de automóvel, dispositivos médicos e até equipamentos

do dia-a-dia como eletrodomésticos, controles remotos, entre outros.

2.3 Plataforma Arduino

O Arduino é uma plataforma eletrônica open source de prototipagem baseada em *hardware* e *software* fáceis de usar [8]. Uma plataforma *open source* é uma plataforma na qual todos os códigos e projetos são desenvolvidos de forma descentralizada e colaborativa, ou seja podem ser distribuídos e modificados, sendo totalmente acessíveis ao público, este modelo descentralizado de desenvolvimento de *software* vai além de uma forma de produção, também é uma maneira inovadora e eficiente de resolver problemas em suas comunidades e setores.

Esta plataforma nasceu como uma ferramenta fácil de rápida prototipagem voltada principalmente para que amadores, estudantes e entusiastas de eletrônica e programação, pudessem desenvolver seus projetos com fácil acesso e baixo custo.

A plataforma Arduino além do baixo custo, também apresenta algumas características interessantes, que são citadas abaixo:

- Portas com conectores que tornam mais fácil a integração com outros componentes.
- Ambiente de fácil desenvolvimento e interação com o usuário que é acessado através de uma porta USB.
- Reguladores de tensão de entrada.
- Saída de alimentação que torna possível ligar outros componentes sem necessitar de outra fonte externa de alimentação.

A Tabela 1 apresenta as especificações de alguns dos principais modelos de placas da família Arduino. Cada modelo apresenta uma especificação única tornando a plataforma Arduino uma plataforma muito versátil.

Tabela 1 – Modelos de placas Arduino [9].

Modelos	Microcontrolador	E/S Digital	E/S Analógica	Clock	Flash
Arduino Uno	ATmega328	14	6	16MHz	32 KB
Arduino Leonardo	ATmega32u4	20	12	16MHz	32 KB
Arduino Mega	ATmega2560	54	16	16MHz	256 KB
Arduino Micro	ATmega32U4	20	12	16MHz	32 KB

Dentre estes modelos citados acima, um deles foi escolhido para o desenvolvimento do projeto e foi melhor abordado no próximo capítulo.

2.3.1 Arduino IDE

O arduino IDE é um ambiente gratuito de desenvolvimento criado pela própria plataforma arduino que facilita o desenvolvimento e a gravação de códigos em C e C++ diretamente no microcontrolador. Outra vantagem do arduino IDE é a versatilidade, este ambiente de desenvolvimento tem compatibilidade com diferentes sistemas operacionais. A Figura 2 apresenta de forma ilustrativa o ambiente de desenvolvimento do arduino IDE.

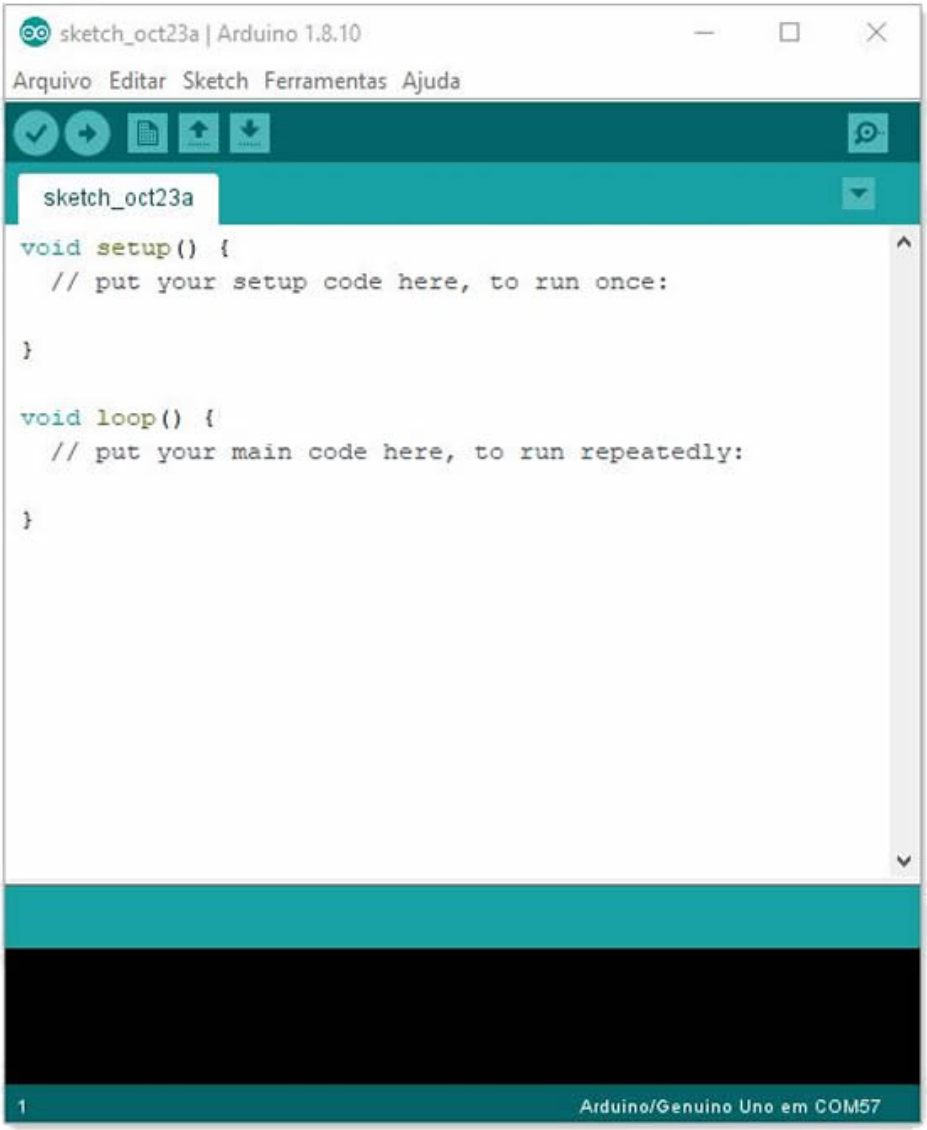


Figura 2 – Imagem ilustrativa do ambiente de desenvolvimento Arduino IDE.

Toda vez que um novo código é gravado no microcontrolador ocorre uma reinicialização do circuito sobrescrevendo os dados na memória. Desta forma a inserção de um novo código no microcontrolador é feito de forma prática e rápida, o que é uma característica vantajosa na hora de testes.

2.4 GPS

O GPS é um sistema de posicionamento global que possui a característica de trabalhar em tempo real, foi desenvolvido pelo departamento de defesa dos Estados Unidos nos últimos anos do século passado. Atualmente a geração de coordenadas geográficas pode ser considerada uma tarefa simples, porém ao longo do desenvolvimento humano sempre procurou-se uma maneira de localização terrestre precisa para substituir as maneiras de orientação pouco precisas proporcionadas pela observação de astros [10].

A ideia inicial do GPS era a de aperfeiçoar o poder das forças armadas dos Estados Unidos, em curto tempo o GPS apresentou grandes avanços para o setor militar, que fez com que essa tecnologia fosse adaptada e desenvolvida também para aplicações comerciais e civis, melhorando o tráfego nos setores de navegação marítima e de aviação, que consequentemente melhoraram o transporte de mercadorias e causou o desenvolvimento do setor comercial. Outro grande avanço foi em relação a otimização de rotas em mapas proporcionando ferramentas que auxiliam a população no dia-a-dia.

Existem dois tipos de serviços de posicionamento para GPS: o padrão SPS (*Standard Positioning Service*) e o mais preciso PPS (*Precise Positioning Service*). O SPS é o serviço de GPS disponível a todos os usuários, enquanto o PPS é um serviço mais preciso, porém é restrito, disponível apenas para uso militar. Nos dias atuais o SPS já apresenta um desenvolvimento tecnológico que proporciona diversas aplicações para os usuários comuns.

Figura 3 – Constelação de satélites GPS.

O GPS é dividido em três segmentos: espacial, controle e de usuários. O segmento espacial é constituído por 24 satélites que estão distribuídos em seis planos orbitais de mesmo tamanho, com quatro satélites em cada plano, conforme é ilustrado na Figura 3. Os satélites são divididos desta forma para garantir que em qualquer lugar da superfície terrestre tenham pelo menos quatro satélites visíveis. O segmento de controle é responsável por monitorar e corrigir os relógios e atualizar periodicamente as mensagens de navegação dos satélites. O segmento de usuário é formado pelos receptores GPS dos usuários. Existem diferentes graus de precisão, dependendo da aplicação pode ser necessário um grau maior ou menor de qualidade de relógio interno do GPS [10].

2.4.1 Funcionamento do GPS

Mesmo o GPS sendo uma tecnologia de extrema sofisticado, a ideia principal do seu funcionamento é relativamente simples. Cada um dos satélites que compõem a formação espacial transmite de forma contínua um sinal de rádio (onda eletromagnética) que contém informações sobre a sua posição orbital e sobre a data e hora de forma precisa, que são marcados por seu relógio atômico interno. Um receptor GPS localizado na terra recebe informações de, no mínimo, quatro satélites diferentes e usa esses dados para calcular sua posição no planeta [11].

Com apenas um receptor GPS é possível obter a localização de qualquer sistema em tempo real e em qualquer lugar do planeta. Os satélites emitem sinais de ondas eletromagnéticas (ondas de rádio) para os receptores, então para determinar a sua posição, os receptores levam em conta o tempo necessário para se comunicar com o satélite e através desse tempo e de um cálculo de determinação de posição, chamado trilateração, fazem com que o receptor calcule a sua própria posição. A localização encontrada é transformada em coordenadas geográficas e então é possível localizar em um mapa a posição atual do receptor.

2.5 Comunicação móvel

Segundo Guimarães [12]: “Pode-se definir como comunicação móvel aquela onde existe a possibilidade de movimento relativo entre partes ou as partes sistêmicas envolvidas. Como exemplo tem-se a comunicação entre aeronaves, entre aeronaves e uma base terrestre, entre veículos, a telefonia celular, a computação móvel, algumas classes de sistemas de telemetria, entre outros.”

Para atingir o objetivo proposto neste trabalho foi necessário a implementação de um sistema constituído por um dispositivo que está se deslocando e ao mesmo tempo consegue enviar e receber dados através de uma rede de internet. Para suprir essa necessidade

foi necessário um serviço de comunicação móvel, portanto neste projeto foi utilizado os serviços de internet móvel GPRS.

2.5.1 Tecnologias GSM e GPRS

A rede GSM, também conhecida como 2G, é uma tecnologia que permite o *roaming* internacional de dados em aparelhos móveis permitindo que o usuário possa utilizar os serviços de *roaming* em diversos países. Atualmente o sistema GSM está presente em boa parte do planeta.

Numa rede GSM, o terminal do utilizador chama-se estação móvel. Um dos componentes da estação móvel é um cartão SIM (*Subscriber Identity Module*), que permite identificar de maneira única este terminal móvel, na maioria das vezes, sendo um telefone móvel [13].

Através do cartão SIM é possível identificar cada usuário, isso é feito através da comunicação entre a estação móvel e uma estação básica (BTS). A comunicação entre essas estações é feita através de uma estação de rádio GSM. Como mostra a Figura 4.

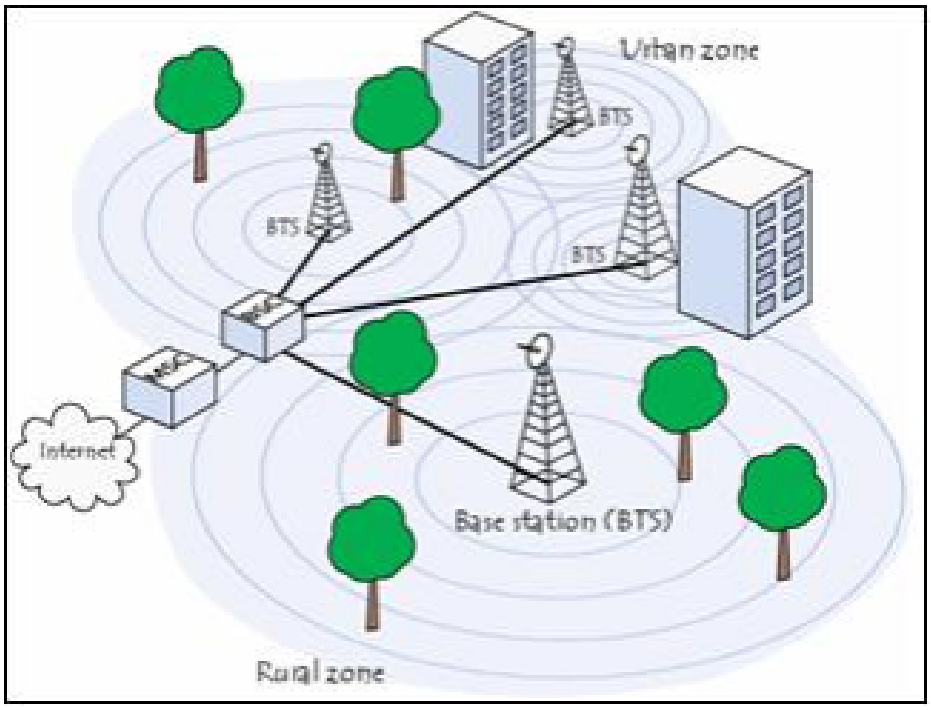


Figura 4 – Arquitetura de uma rede GSM [13].

O GPRS é uma tecnologia que representa a evolução do GSM, através do GPRS foi possível introduzir a transferência de dados por pacotes através do protocolo IP, em outras palavras utilizar o serviço de internet em um terminal móvel. A grande vantagem deste serviço é que neste caso as transmissões de dados utilizam a rede apenas quando solicitado. O GPRS apresenta um formato em que a transmissão de dados é feita apenas quando os dados são utilizados, conhecido como comutação de dados por pacotes, diferente do

GSM que a transmissão de dados é feito de forma contínua, conhecido como comutação de dados por voz. Portanto o GPRS além de apresentar vantagem de maior velocidade de conexão, também é uma solução mais eficiente em relação ao GSM.

A Figura 5 representa de forma simples como funcionam os dois serviços, GSM, comutação de dados por voz, e GPRS, comutação de dados por pacote. O protocolo RTCP (*Real-Time Transport Control Protocol*) representa a rede para comunicação de voz, enquanto o elemento IP (*Internet Protocol*) representa o ponto de integração com a Internet.

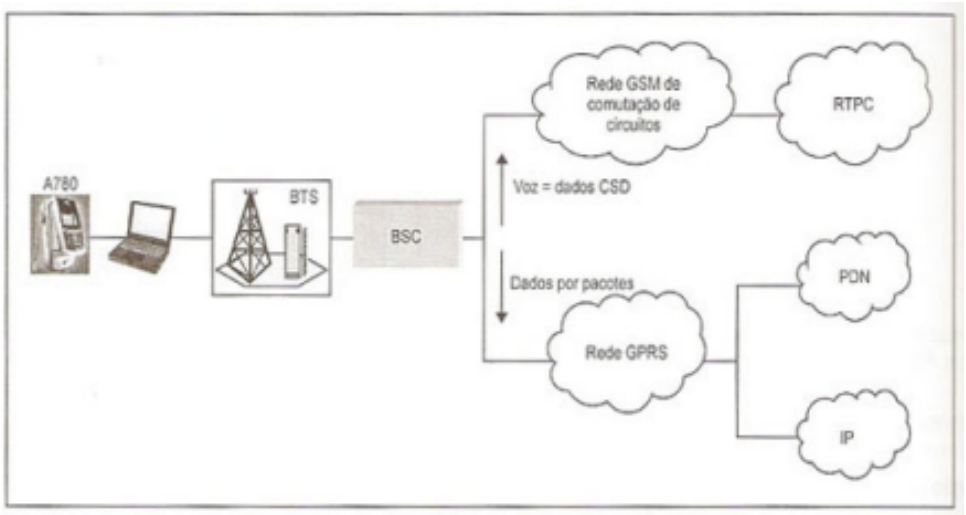


Figura 5 – Comutação de pacotes nas redes GSM e GPRS.

Com o novo sistema de transmissão de dados por pacotes e o aumento na taxa de transmissão, o GPRS proporcionou a utilização de chats com vários usuários, compartilhamento de imagens, transferência de documentos escritos, envio e recebimento de e-mails sem verificação no servidor, entre outras melhorias.

2.5.2 Tecnologias 3G e 4G

Nos anos 2000, começaram a surgir de fato os serviços de Internet nos celulares, isso ocorreu através da tecnologia 3G, essa conexão permitiu transformar celulares em *smartphones*. A Terceira Geração (3G) é a ascensão das tecnologias de segunda geração, GSM (2G) e GPRS (2.5G), e se caracteriza pela oferta de banda larga sem fio. Os serviços de tecnologias 2G por disponibilizar apenas enviar mensagens e realizar chamadas não estavam mais suprindo as necessidades dos usuários. Visando melhorar esses serviços, a 3G foi criada para permitir além de acesso à Internet, transmissão de dados de voz e serviços de multimídia muito mais eficientes, ou seja, a partir da 3G foi possível em aparelhos móveis utilizar serviços de e-mail online, navegação *web*, *download* e compartilhamento de vídeos de baixa resolução, compartilhamento de imagens, entre outras aplicações.

Os serviços 3G oferecem melhores taxas de transmissão por faixa de frequência em relação ao 2G, proporcionando velocidades de transmissão de até 2 Mbps juntamente com

menor custo tanto para a empresa prestadora do serviço como para o usuário. Também foram adicionadas melhorias na parte de segurança das redes, com a tecnologia 3G ocorreu a adição de novas técnicas de autenticação, através de algoritmos e utilização de chaves de segurança maiores.

Ao entrar no tema de tecnologia 4G é necessário abordar o que é o protocolo IP. O protocolo IP nada mais é do que o protocolo de comunicação utilizado na internet, este protocolo emprega a abordagem de transporte de dados através de pacotes. A tecnologia 4G passou a ser totalmente baseada no protocolo IP, coisa que nas gerações anteriores não era possível, isso proporcionou um grande avanço para a comunicação móvel.

A tecnologia de Quarta Geração(4G) surgiu em 2010 e apresentou um conceito que vai muito além de telefonia móvel. Esta tecnologia proporciounou uma convergência entre as redes de cabo e sem fio, diferente das tecnologias desenvolvidas até o 3G. Ela permite a comunicação de voz, vídeo e dados de forma bilateral, proporcionando melhor qualidade nos serviços, mesmo que o usuário esteja em movimento, com essa tecnologia, o seu sinal de Internet não será reduzido de forma significativa, sendo possível utilizar os mesmos serviços, coisa que não era possível nas tecnologias 3G.

A tecnologia 4G apresenta como principal avanço a possibilidade de utilizar altas taxas de transferências de dados. Esta tecnologia é totalmente baseada no protocolo IP, sendo então composta por redes de celulares, redes sem fio(*wireless*), computadores e dispositivos eletrônicos [14].

A tecnologia 4G possibilitou utilizar os dados móveis para realizar serviços que antes eram limitados apenas a conexões fixas. Ela permitiu a reprodução sem travamento de vídeos em alta resolução, videoconferências sem instabilidade, transmissões ao vivo online, entre outras aplicações que não eram possíveis com a tecnologia 3G.

2.6 Comando AT

O conjunto de comandos AT (abreviatura da palavra *Attention*), também conhecido como conjunto de comandos *Hayes*, foram desenvolvidos em 1971 por *Dennis Hayes* e implementados para o modem *Hayes Smartmodem* em 1981. O objetivo desta linguagem era criar uma interface para configuração e fornecimento de instruções a partir de comandos.

Inicialmente os comandos AT foram criados para fornecer comunicação entre *modems*. Com a evolução da telefonia celular e a criação do GSM, estes comandos passaram a ser utilizados como padrão de comunicação, permitindo que fossem realizadas ações como, chamadas telefônicas, transmissão de dados, SMS, entre outros serviços [15]. O comando AT também pode ser utilizado para fazer a comunicação de microcontroladores com outros hardwares, como módulos.

Os comandos consistem em uma série de cadeias de texto curtos que podem ser combinados e assim produzir serviços de chamada, como, enviar e receber SMS, auto-respostas e verificações de qualidade de sinal e conexão.

Alguns dos principais comandos disponíveis são:

- AT - Testa a resposta do módulo.
- AT+CIFSR - Pega o endereço de IP local.
- AT+CIPSTART - Estabelece conexão TCP, transmissão UDP ou conexão SSL.
- AT + CIPSEND - Envia dados ao servidor remoto.
- AT+CIPMUX=0 - Ativa o modo texto.
- AT+CGATT=1 - Habilita o serviço de Internet do cartão SIM.
- AT + CSTT - Configura o APN do cartão SIM.

2.7 Webservices

Segundo Tamae [16]: "*WebServices* são conjuntos de aplicações autodescritivas que podem ser publicadas, localizadas e invocadas através da *Web*. Estas aplicações podem ser desde simples processos como troca de mensagens a complexas transações comerciais ou industriais como um processo de compra de mercadorias. Uma vez que um *WebServices* é publicado, outras aplicações (ou até mesmo outros *WebServices*) podem acessá-los e invocá-los, tanto para obtenção de dados como para a interação com os serviços que uma organização oferece."

De forma geral, um *webservice* através de uma rede faz com que os recursos de uma aplicação estejam disponíveis para que outras aplicações, através do protocolo HTTP, possam acessá-las, esses recursos são identificados separadamente por um URL (*Uniform Resource Locator*). *WebServices* apresenta uma forma de comunicação padronizada e independente entre os serviços, possibilitando utilizar qualquer tipo de sistema operativo, plataforma de hardware ou linguagem de programação de suporte *Web*. Dessa forma o *WebServices* é uma tecnologia ideal para fazer a comunicação entre aplicações através da Internet de forma flexível, prática e segura.

2.7.1 Rest

Representational State Transfer (REST), em português Transferência Representacional de Estado, é um estilo de arquitetura de *software* que define um conjunto de restrições a serem usadas para a criação de *web services* (serviços Web). Os *Web services* que utilizam o estilo de arquitetural REST, são denominados *Web services RESTful* e possuem

a capacidade de se comunicar de forma transparente com outro sistema através da internet. Os *Web services RESTful* permitem aos sistemas acessar e manipular representações textuais de recursos da *Web* usando um conjunto predefinido de operações sem estado [17]. A arquitetura REST apresenta como protocolo de comunicação o HTTP, que é um protocolo muito utilizado para comunicação *web* e tem como uma de suas característica a transmissão de documentos de hipermídia, como o HTML.

Na arquitetura REST afirma-se que a Web fornece escalabilidade através de uma série de conceitos de projeto fundamentais:

- Um protocolo cliente/servidor sem estado: As mensagens enviadas entre o cliente e o servidor apresentam todas as informações necessárias para o envio e recebimento destas mensagens. O cliente e o servidor não necessitam gravar os estados das comunicações entre as mensagens. Muitas aplicações baseadas no protocolo HTTP utilizam *cookies* para manter o estado da sessão.

- Um conjunto de operações bem definidas que se aplicam a todos os recursos de informação: O protocolo HTTP possui um conjunto de operações, sendo os mais importantes, POST, GET, PUT e DELETE. Muitas vezes essas operações do protocolo HTTP são combinadas com operações do tipo CRUD para manipulação de bancos de dados, neste caso o POST não é utilizado.

- Uma sintaxe universal para identificar os recursos. No sistema REST, cada recurso é unicamente direcionado através da sua URI.

- Uso de hipermídia: A hipermídia é utilizada para informação da aplicação e para transição de estado da aplicação. A representação de hipermídia em arquiteturas REST são geralmente do tipo HTML ou XML. Isso permite através da arquitetura REST navegar a outros recursos seguindo ligações, não sendo necessário o uso de registros ou outra infraestrutura adicional.

Um conceito característico da arquitetura REST é a existência de recursos (elementos de informação), esses recursos podem ser manipulados através de um identificador global. A troca de recursos, que geralmente são do tipo XML ou JSON, ocorre entre os componentes da rede (clientes e servidores) e a comunicação é feita através da interface padrão (HTTP). Assim, uma aplicação pode interagir com um recurso através apenas do identificador do recurso e da ação requerida, não sendo necessário entrar em contato com outras informações que estão guardadas no servidor.

2.7.2 Protocolo MQTT

Protocolos de rede são os conjuntos de normas que permitem que duas ou mais máquinas conectadas à Internet se comuniquem entre si. Funciona como uma linguagem

universal, que pode ser interpretada por computadores de qualquer fabricante, por meio de qualquer sistema operacional.

A IoT está em rápida evolução e possui alguns protocolos de comunicação, dentre eles, o protocolo MQTT (*Message Queue Telemetry Transport*) tem sido adotado por diversas empresas, tanto para uso em sistemas IoT ou apenas como um protocolo de comunicação [18].

O MQTT é um protocolo que foi desenvolvido inicialmente pela IBM, mas em pouco tempo passou a ser um padrão aberto para comunicação entre dispositivos. Este protocolo está situado na camada de aplicação da arquitetura TCP/IP e utiliza o padrão de mensagens do tipo *publisher/subscriber* (publicador/assinante), neste caso todos os dados são enviados para um intermediário, chamado *broker*, que tem a função de enviar as mensagens aos destinatários corretos. Esse tipo de estrutura permite que o textitpublisher e o *subscriber* se comuniquem sem a troca de endereços entre as partes, apenas o endereço do *broker* precisa ser conhecido. Esta estrutura permite três tipos de relacionamento, *One-to-one, One-to-many e Many-to-many* conforme mostra a Figura 6.

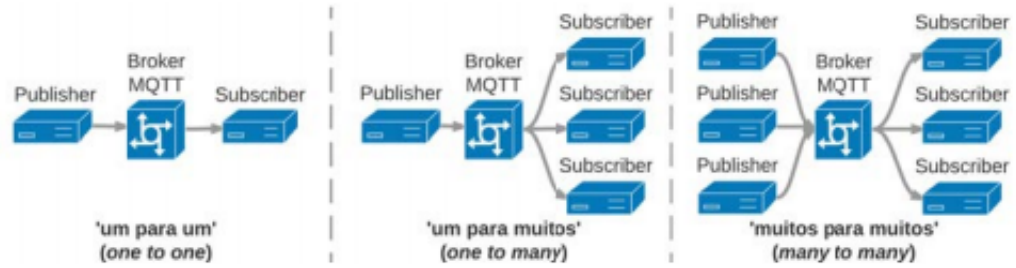


Figura 6 – Tipos de distribuição de mensagem suportados pelo protocolo MQTT [18].

- One-to-one: O relacionamento um-para-um é onde as duas partes contém apenas uma instância, ou seja um *publisher* só pode se comunicar com um único *subscriber* por vez, e vice-versa.
- One-to-many: Neste relacionamento, um-para-muitos, o *publisher* pode se comunicar com vários *subscribers* de uma vez, porém os *subscribers* podem apenas se comunicar com aquele único *publisher*.
- Many-to-many: O *many-to-many* (muitos-para-muitos) é o relacionamento em que a comunicação entre as duas partes é completamente livre, neste caso vários *publishers* podem se comunicar com vários *subscribers*, assim como podem também se comunicar apenas um com um.

Como o MQTT usa um *broker* para a comunicação entre as partes, além de vantagens relacionadas a segurança também apresenta baixo uso de banda e baixo uso de processamento e memória, tornando este protocolo muito comum em sistemas que utilizam IOT.

3 Revisão de Literatura

Durante o levantamento bibliográfico realizado na procura de projetos relacionados foram encontrados diversos projetos de extrema relevância, dentre eles três foram selecionados por apresentarem ideias interessantes sobre o uso de microcontroladores, GPS, GSM e integração de sistemas.

3.1 Protótipo de um sistema de rastreamento veicular baseado no módulo Telit

Em [19] é apresentado um projeto de sistema de rastreamento veicular baseado no módulo Telit, além dos dispositivos para obtenção das coordenadas geográficas o sistema também é composto por uma câmera de segurança instalada junto aos outros componentes. Os principais objetivos apresentadso neste projeto são obter as coordenadas geográficas do automóvel e armazenar uma imagem do interior do veículo que será captura no instante do fornecimento da localização, em seguida os dados são disponibilizados em uma página web junto com um botão que possibilita o bloqueio do veículo.

A Figura 7 apresenta o sistema proposto em [19], percebe-se que neste sistema fez-se o uso de vários *hardwares* como, sensores, um *display* LCD, uma câmera, um relé, um módulo Telit, entre outros.

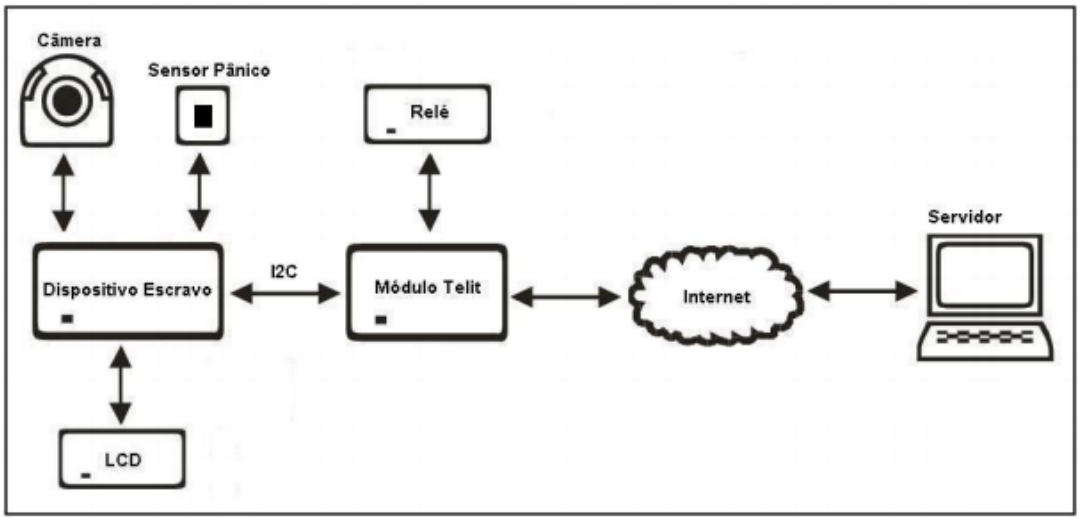


Figura 7 – Sistema para rastreamento veicular via Telit [19].

Este trabalho tem como foco uma abordagem prática com maior envolvimento no desenvolvimento do hardware, apresentando funcionalidades bem relevantes como efetuar bloqueio do veículo de forma remota, além da obtenção de imagens internas do veículo e

de suas coordenadas geográficas. Entretanto como o foco era maior no desenvolvimento do hardware o projeto não apresenta obtenção de dados em tempo real, os dados do veículo só são obtidos apenas quando solicitados e a forma de apresentação desses dados é feita em um ambiente *web* simples, sem muita interação com o usuário.

3.2 Implementação do Sistema de Mapeamento de uma Linha de Ônibus para um Sistema de Transporte Inteligente

Segundo Weigand et al. [20], o usuário do transporte coletivo urbano por ônibus no Brasil, na grande maioria das vezes não sabe ao certo o horário em que deve chegar às paradas e nem os horários exatos dos ônibus. Nota-se, então, a evidente necessidade de um sistema de informações sobre horários, que beneficie o usuário de transporte urbano coletivo por ônibus.

Visto isso, Weigand et al. [20] desenvolveu um trabalho com o objetivo de produzir um sistema de informações de horários para usuários e administradores de transporte coletivo urbano. A Figura 8 apresenta como funciona o sistema desenvolvido por Weigand et al., percebe-se que o sistema apresenta tecnologias de telefonia celular, GPS e comunicação de equipamentos.

Figura 8 – Diagrama do sistema de mapeamento de linha de Ônibus [20].

O sistema proposto em [20] tem foco em um problema não voltado à segurança, porém também utiliza redes de telefonia celular, módulo de GPS e integração de sistemas para coleta e envio de dados. Neste caso o grande foco deste trabalho é o algoritmo a ser

desenvolvido para que da melhor forma possível apresente uma programação de horários de ônibus que satisfaça tanto os usuários como os funcionários.

3.3 Rastreador veicular via WEB

Em [21] é apresentada uma proposta de um sistema de rastreador veicular via WEB, neste projeto foi apresentado diversos componentes separados conectados através de eletrônica básica. O sistema é composto por módulo GPS, módulo GSM, microcontrolador, um dispositivo de gravação, antenas de GPS e GSM, todos esses dispositivos são conectados através de uma *protoboard* e alguns *jumpers* juntamente com alguns componentes de eletrônica necessários, como resistores, capacitores entre outros, como mostra a Figura 9. Os dados são obtidos e enviados para um servidor WEB no qual Renato desenvolveu uma interface simples para visualização.

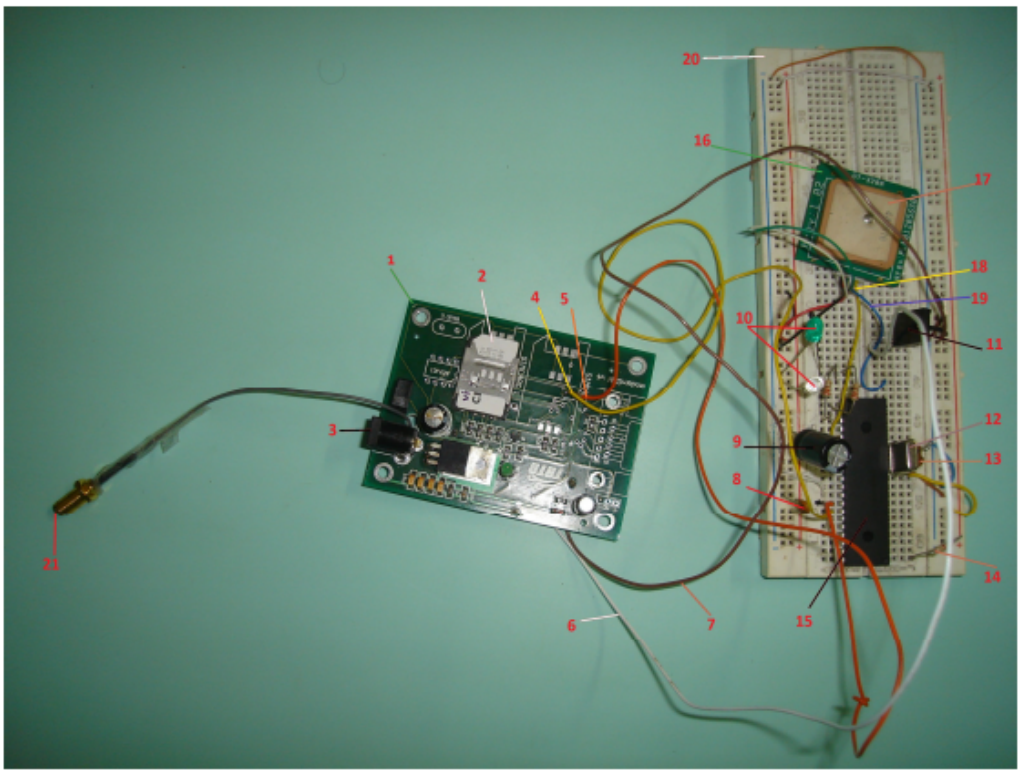


Figura 9 – Protótipo para rastreamento via Web [21].

Este projeto apresenta uma abordagem mais antiga, neste caso foi utilizado uma série de componentes que hoje em dia poderiam ser substituídos por outros equipamentos mais poderosos facilitando o desenvolvimento da parte de hardware e obtendo dados de forma mais rápida. Outro ponto importante é que neste trabalho o autor desenvolveu um servidor *Web* para armazenamento dos dados ao invés de utilizar uma webservice que é uma forma mais simples e eficiente de armazenamento.

4 Desenvolvimento

Este capítulo tem por objetivo expor de forma detalhada todos os passos utilizados para implementação completa do sistema proposto, primeiro será mostrado as especificações dos componentes e as implementações necessárias para o funcionamento correto do hardware e do software desenvolvido, em seguida será explicado a forma que foram feitas as comunicações com o *webservice* e por fim será exibido todas as etapas e ferramentas utilizadas para o desenvolvimento completo do aplicativo.

4.1 Funcionalidades do sistema

Inicialmente para o desenvolvimento do projeto proposto é necessário definir uma ideia base e os elementos que serão utilizados para compor o sistema. O projeto é composto por alguns elementos que recebem e emitem dados, para se chegar até o resultado desejado segue-se uma série de etapas:

- O automóvel a ser verificado deve possuir um módulo embarcado instalado que será responsável por obter as coordenadas de localização geográficas, esses dados são obtidos através da tecnologia GPS que será parte do sistema embarcado.
- Após a obtenção dos dados, eles são enviados para um serviço Web, para fazer este envio foi escolhido o uso de uma rede GPRS, esse serviço de rede é acessado através de um SIM Card de uma operadora de telefonia móvel.
- O serviço Web recebe as informações através de uma solicitação GET e armazena esses dados em uma base de dados que podem ser visualizadas através de um painel.
- O usuário então poderá fazer o acesso às informações do sistema através de um aplicativo desenvolvido para sistemas Android. Este aplicativo oferece uma interface com acesso a algumas informações, como a latitude e longitude do automóvel junto com a localização em um mapa e um sistema de rotas que apresenta a melhor rota entre o usuário e o automóvel. Também é disponibilizada uma funcionalidade que permite ao usuário, dentro do aplicativo, utilizar os serviços do Google Maps com todas as ferramentas deste serviço disponíveis.

A Figura 10 representa de forma ilustrativa como funciona o sistema.

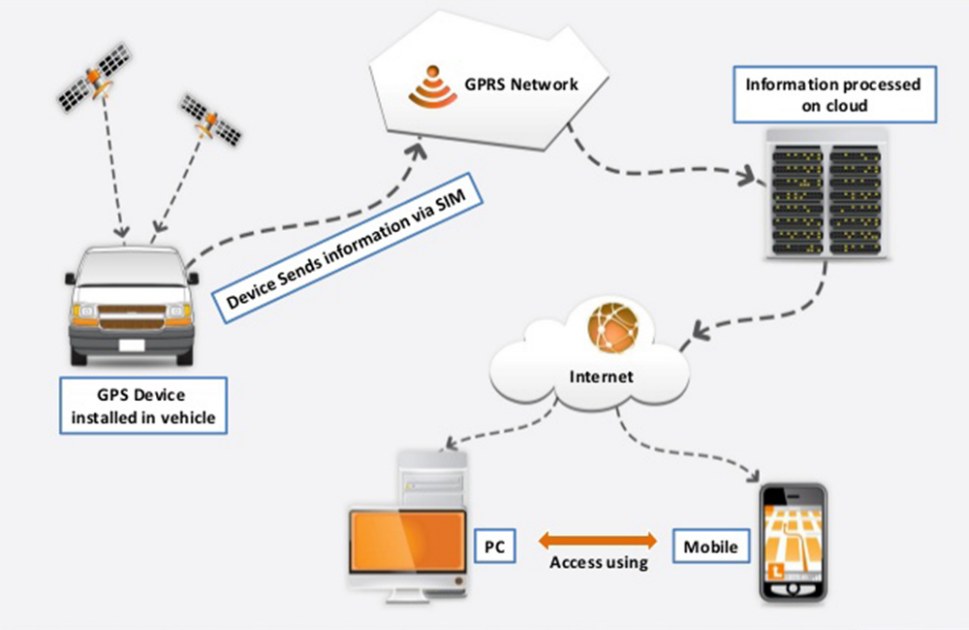


Figura 10 – Sistema para rastramento veicular via GPS [22].

4.2 Hardware

Para implementação do projeto foi escolhida a plataforma de prototipagem Arduino, citada no Capítulo 2. O desenvolvimento deste projeto necessita de alguns hardwares, dentre eles, uma placa controladora, um módulo de GPS, um módulo GSM/GPRS e um cartão SIM. Com isso dentre os possíveis hardwares pesquisados foram escolhidos para compor o sistema:

- Placa controladora Arduino Uno: Dentre a grande quantidade de placas disponibilizadas pela plataforma Arduino, optou-se por utilizar o Arduino Uno. Essa placa foi escolhida por apresentar um custo econômico relativamente baixo e suas especificações suprirem as necessidades do projeto, além de vários outros projetos acadêmicos que envolvem sistemas embarcados optarem por essa placa. devido a facilidade que ela apresenta de conectar a maioria dos *shields* e módulos disponíveis e possuir documentação disponível na Internet.

- Módulo SIM 808: Para os serviços de GPS e GSM foi escolhido o módulo SIM 808. Este módulo é composto por um módulo de GPS e um módulo de GSM integrados em uma única placa. Dentre a enorme variedade de módulos GPS e GSM existentes, a SIM 808 foi selecionado devido a sua enorme praticidade em conectar o GPS e GSM e por apresentar facilidade de comunicação com o Arduino Uno, além de uma biblioteca própria que auxilia na hora da implementação do código.

- Cartão SIM: O cartão SIM escolhido foi um básico que possui serviço de dados pré-pago de uma empresa de telefonia, este cartão apresenta o necessário para utilizar

os serviços do GPRS.

4.2.1 Arduino UNO

Arduino Uno é uma placa microcontrolada baseada no ATmega328. Possui 14 pinos de entrada/saída digital (dos quais 6 podem ser usados como saídas PWM), 6 entradas analógicas, um ressonador de cerâmica de 16MHz, uma conexão USB, um conector de alimentação, um conector ICSP e um botão de reinicialização . Ele contém tudo o que é necessário para dar suporte ao microcontrolador, o usuário precisa simplesmente conectar a placa a um computador com um cabo USB ou ligá-lo com uma fonte de energia para começar a programá-la [23].

Esta placa também possui um microcontrolador, ATMEL ATMEGA16U2, que possibilita o *upload* binário do código feito pelo usuário. Este microcontrolador também possui dois leds (TX, RX) que indicam quando ocorre o envio ou o recebimento dos dados da placa para o computador, além de permitir que a placa entre no modo *bootloader* automaticamente após o *upload* do novo código.

A Figura 11 representa uma imagem do Arduíno Uno onde é possível localizar todos os componentes citados anteriormente.

Figura 11 – Placa Arduino Uno.

4.2.2 Especificações do arduino UNO

A Tabela 2 apresenta de forma mais detalhada algumas especificações da placa Arduino Uno.

Tabela 2 – Tabela de especficações do arduino uno [23].

Microcontrolador	ATmega328
Tensão operacional	5V
Tensão de entrada (recomendado)	7-12V
Tensão de entrada (limite)	6-20V
Pinos de E / S digitais	14 (dos quais 6 fornecem saída PWM)
Pinos de E / S digital PWM	6
Pinos de entrada analógica	6
Corrente DC por pino de I / O	20 mA
Corrente DC para pino de 3,3 V	50 mA
Memória flash	32 KB
SRAM	2 KB
EEPROM	1 KB
Velocidade do clock	16 MHz
Comprimento	68,6 mm
Largura	53,4 mm
Peso	25 g

4.2.3 Alimentação do Arduino UNO

A placa apresenta a possibilidade de alimentação através de conexão USB ou por fonte de alimentação externa. A alimentação externa é feita através de um conector *Jack* e o valor da tensão deve estar entre 6-20 volts, porém não é recomendado a alimentação com tensões inferiores a 7V por apresentar chance de instabilidade e nem superiores a 12V que neste caso o regulador de tensão da placa pode superaquecer e danificar a placa [24].

Já na alimentação por conexão USB assim que o cabo é conectado a placa é alimentada diretamente pelo USB, então a tensão não precisa ser estabilizada pelo regulador de tensão. O circuito do USB apresenta componentes que garantem a proteção da porta em caso de anormalidades relacionadas a tensão.

O Arduino Uno também apresenta alguns conectores de alimentação que podem ser usados para conectar *shields* e módulos a placa, que são:

- 3,3 V: Fornece uma tensão estabilizada de 3,3V com corrente máxima de 50mA, pode ser utilizado para alimentação de shields, módulos e circuitos externos.
- 5 V: Fornece uma tensão estabilizada de 5V, pode ser utilizado para alimentação de shields, módulos e circuitos externos.
- GND: Pino terra, ou seja um pino de referência.
- VIN: Este pino é usado quando utiliza-se a alimentação externa através do conector *jack*, a tensão do pino será a mesma da fonte externa.

4.2.4 Pinagem do Arduino UNO

Como mostrado em seções anteriores, a placa Arduino Uno possui pinos de entrada e saída digitais e pinos de entrada analógicos. A Figura 12 representa um esquemático do Arduino Uno onde é possível localizar cada pino na placa.

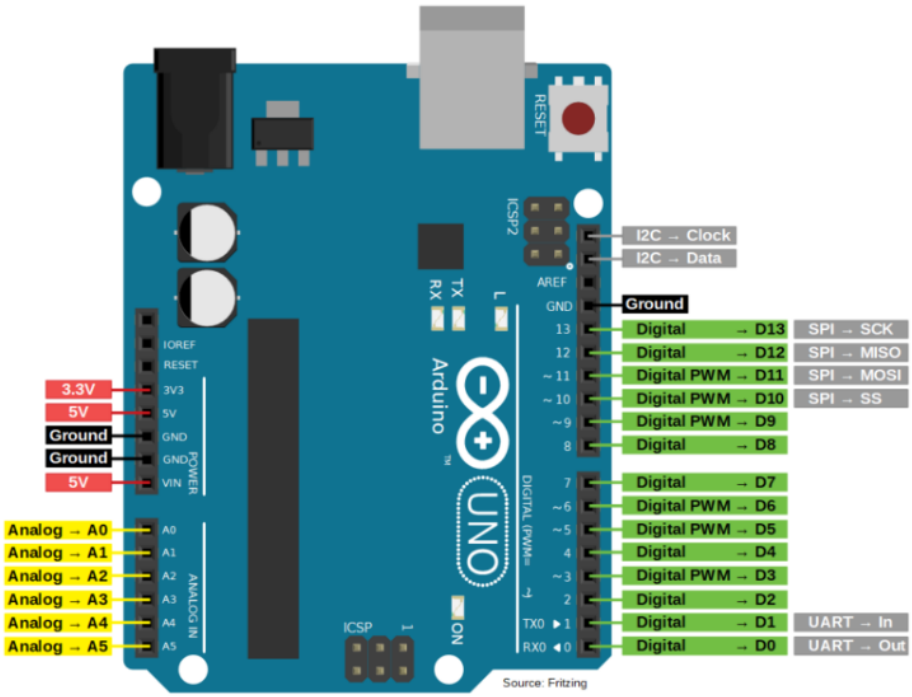


Figura 12 – Pinagem do Arduino Uno [25].

Conforme visto na Figura 12 pode-se observar que o Arduino Uno possui 14 pinos digitais que podem ser usados como entrada ou saída, estes pinos operam em 5 V e podem fornecer ou receber uma corrente máxima de 40 mA. Essas portas digitais só podem assumir dois estados, *HIGH* ou *LOW*, ou seja, 0 V ou 5 V.

A manipulação desses pinos digitais pode ser feita através de algumas funções que são habilitadas por software, sendo elas:

- pinMode(): Configura um pino específico para que se comporte como entrada ou como saída.
- digitalWrite(): A função digitalWrite é usada para definir o estado em um pino digital, sendo *HIGH* para 5V ou *LOW* para 0V(terra).
- digitalRead(): Esta função lê o valor de um pino digital específico e retorna o seu estado sendo *HIGH* ou *LOW*.

Alguns desses pinos também possuem funções especiais, sendo algumas delas:

- PWM: Os pinos 3,5,6,9,10 e 11 podem ser usados como saídas PWM de 8 bits através da função analogWrite().
- Comunicação serial: Os pinos 0 e 1 podem ser utilizados para comunicação serial. Esses pinos são usados para a troca de dados entre o microcontrolador e o computador que estão conectados via USB.
- Interrupção externa: Através dos pinos 2 e 3 é possível configurar uma interrupção externa ao sistema através da função attachInterrupt().
- LED: O pino 13 possui um LED conectado a ele, quando este pino está no valor *high* o LED está aceso, quando o pino está em *low* o LED está apagado.
- SPI: Os pinos 10(SS), 11(MOSI), 12(MISO) e 13(SCK), através da biblioteca SPI podem ser utilizados para comunicação SPI.

Além das portas digitais, o Arduino Uno possui 6 portas analógicas que possuem a resolução de 10 bits cada, essas portas são nomeadas de A0 a A5, como pode-se observar na Figura 12. Para a manipulação das portas analógicas existem apenas duas funções desenvolvidas pela plataforma arduino, que são:

- analogReference(): Através da função analogReference() é possível configurar a tensão de referência para as entradas analógicas.
- analogRead(): Através da função analogRead() é possível ler o valor presente em um pino especificado. A máxima frequência de leitura que se pode ter é de 10.000 vezes por segundo.

4.2.5 Programação do Arduino UNO

A programação do microcontrolador será feita em linguagem C/C++ através da Arduino IDE que foi abordada na Seção 2.3.1. O Arduino Uno vem programado com o programa inicializador *bootloader* o que possibilita o upload de novos códigos sem uso de hardware para programação externa.

Outra opção de programação seria através de programação com hardware externo, isso é feito utilizando o conector ICSP com um circuito programador ATMEL que permite programar o microcontrolador usando a porta serial e um programa especial para essa função.

4.2.6 Módulo SIM 808

O módulo SIM 808, é um módulo projetado para o mercado global. Ele é integrado a um chip GSM/GPRS de alto desempenho e também possui um motor GPS e um motor BT [26].

O SIM 808 possui baixo consumo de energia em modo de hibernação e pode ser integrado com um circuito de carregamento de baterias de lítio, o que possibilita um cenário conveniente para projetos que necessitem estar ligados constantemente. Possui alta sensibilidade de recepção de GPS com 22 canais de recepção de rastreamento e 66 de aquisição. Além disso, também suporta A-GPS que está disponível para localização em interiores. O módulo é controlado pelo comando AT via UART e suporta níveis lógicos de 3,3V e 5V [27].

Para construção do projeto, inicialmente foi conectado os módulos de GPS e GSM/GPRS a placa do SIM 808 conforme mostra a Figura 13.

Figura 13 – SIM808 com os módulos conectados.

Após conectar os módulos a placa, o SIM 808 está preparado para ser conectado ao Arduino UNO, isso é feito através da conexão de três *jumpers*. Primeiro deve-se conectar um dos *jumpers* entre GND do arduino e o GND da SIM 808, em seguida conectar um *jumper* no pino RX e outro no pino TX do SIM 808 e então ligar esses dois *jumpers* aos pinos digitais do Arduino UNO definidos no código. No caso deste projeto foi conectado o RX ao pino 10 e o TX ao pino 11, como é ilustrado nas Figuras 14.

A Figura 15a representa o Arduino UNO e o módulo SIM808 já conectados e prontos para serem utilizados. Para realização dos testes a alimentação do Arduino UNO foi feita por um cabo USB e a do SIM 808 pode ser feita por uma fonte de 5 a 20V, neste caso foi utilizada uma fonte externa de 12V. Como mostra a Figura 15b. No caso de testes em veículos será necessário utilizar uma báteria para alimentação do SIM 808.

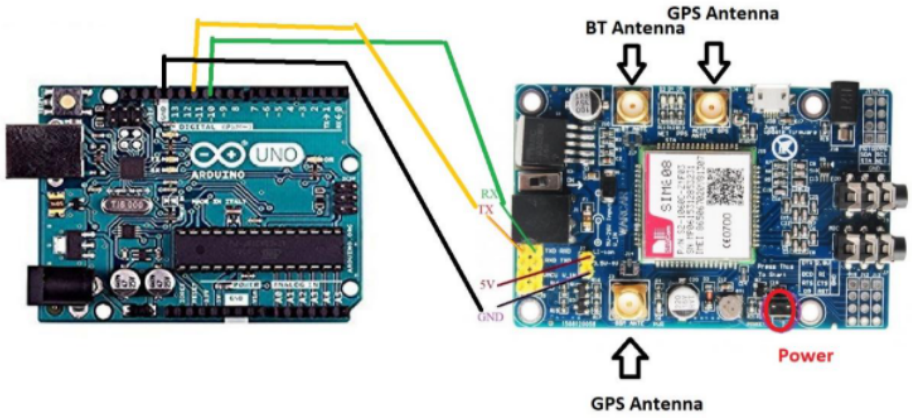
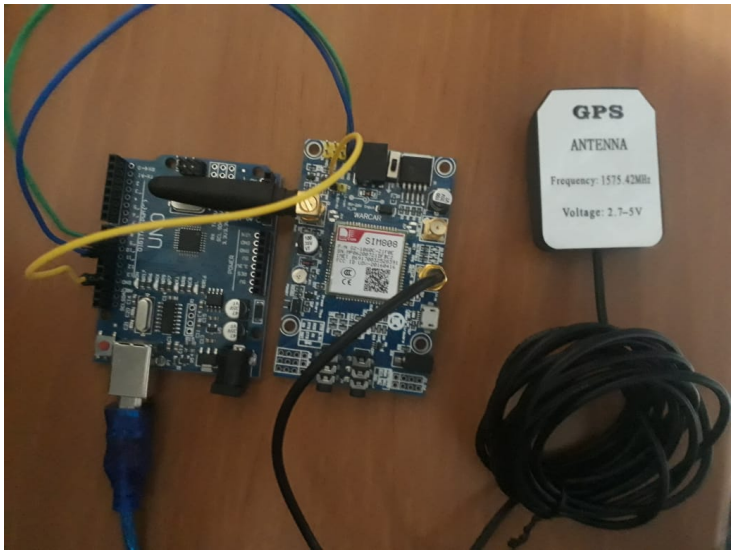


Figura 14 – Imagem ilustrativa das conexões entre o Arduino UNO e o SIM808 [26].



(a) Arduino UNO e SIM808 conectados.

(b) Fonte de 12V.

Figura 15 – Arduino, módulos e fonte.

Após a montagem toda vez que a SIM 808 for conectada a fonte deve-se ligar os módulos de GPS e GSM/GPRS através dos botões da placa, é possível perceber quando os módulos estão ligados através dos *leds* acesos, como mostra a Figura 16.

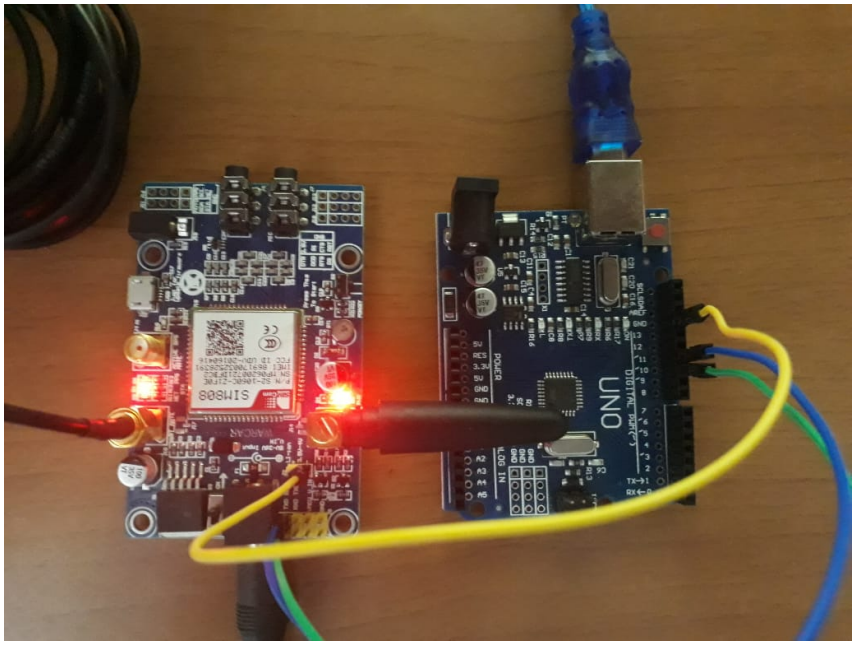


Figura 16 – SIM808 ligada.

4.2.7 SIM Card

Para o funcionamento correto do sistema é necessário inserir um cartão SIM, a Figura 17 representa o cartão utilizado no projeto.

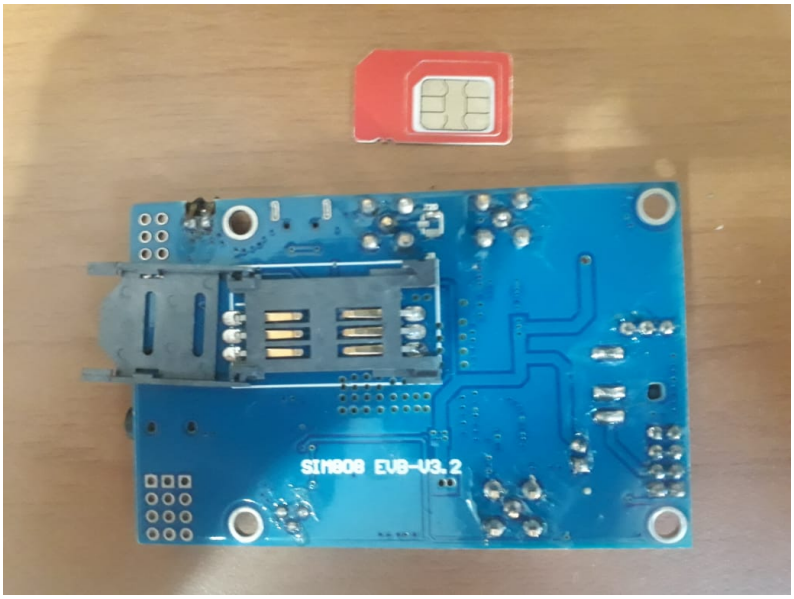


Figura 17 – SIM Card e SIM808.

O cartão SIM é responsável por possibilitar que seja utilizado os dados móveis de rede pelo módulo de GSM/GPRS. O cartão SIM é inserido na placa do módulo SIM 808 na parte de trás conforme mostra a Figura 18.

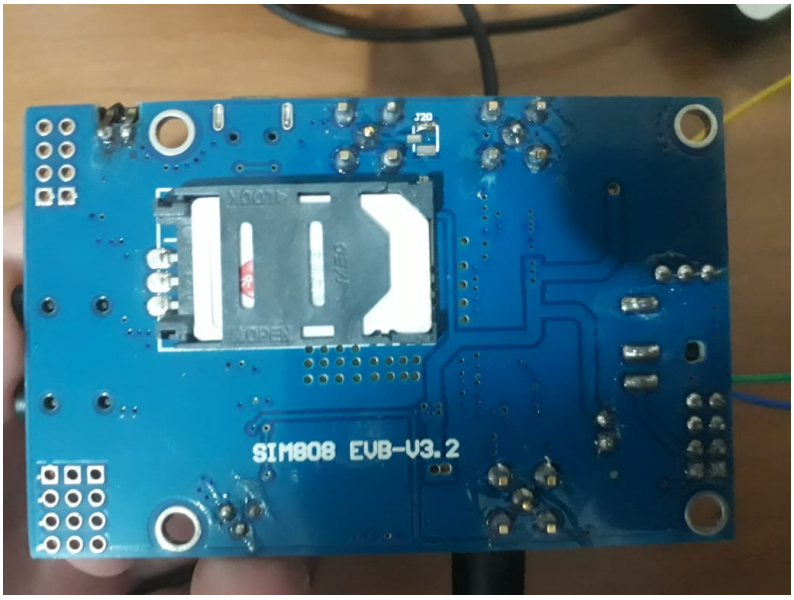


Figura 18 – SIM Card conectado a SIM808.

Após a inserção do cartão SIM o módulo SIM 808 já está preparado para enviar e receber dados através do GSM/GPRS.

4.3 Desenvolvimento do Firmware

O desenvolvimento do *firmware* que será gravado na memória flash do Arduino foi realizado em linguagem C na IDE do Arduino como foi descrito na Seção 3.3.5. Este *firmware* apresenta como principais funções:

- Comunicação entre o microcontrolador Arduino Uno e o módulo SIM808.
- Obtenção de dados pelo GPS.
- Conexão entre o Arduino Uno e o GPRS.
- Envio dos dados para o *webservice* através do GPRS.

O modelo padrão de programação na plataforma Arduino IDE é feito a partir de duas funções principais obrigatórias:

- void setup(): Esta função só é executada uma vez, quando o microcontrolador é ligado, é utilizada para declaração de variáveis, bibliotecas e configuração de pinos.
- void loop(): Esta função é executada após o setup e diferente dele é executado repetidas vezes enquanto o microcontrolador permanecer ligado, funciona como um laço de repetição. É utilizado para realizar de fato a atividade desejada pela placa controladora.

O *firmware* desenvolvido funciona primeiro executando a função de setup, nesta função é onde ocorre as conexões via *software* entre o módulo SIM 808 e o Arduino Uno (através das portas *serials*), os testes de conexão do SIM808 e o teste de energia do GPS. Em seguida é executado a função loop, esta função só é iniciada após a função setup ser encerrada, no loop inicialmente procura-se sinal de área para o GPS, após o GPS conseguir encontrar sinal, parte-se para a obtenção dos dados e envio para o *webservice*, esta função é executada continuamente enquanto a microcontrolador permanecer ligado, ou seja os dados são obtidos e enviados continuamente até a placa ser desligada ou o módulo desconectado.

A Figura 19 representa alguns resultados da execução do *firmware*, obtidos do *serial monitor* do Arduino IDE.

```
sim808 inicializado com sucesso.
GPS ligado.
GPS procurando sinal...


GPS encontrou sinal.
Novos dados obtidos:
Data:27/4/2021 19:20:52:0
latitude :-26.928455
longitude :-49.065120
http://maps.google.com/maps?q=-26.928455,-49.065120
speed_kph :0.17
heading :13.83

GET https://api.thingspeak.com/update?api_key=4RX475FEXIOKAXHW&field1=-26.928455&field2=-49.065121&field3=0.17


GPS encontrou sinal.
Novos dados obtidos:
Data:27/4/2021 19:21:14:0
latitude :-26.928815
longitude :-49.064720
http://maps.google.com/maps?q=-26.928815,-49.064720
speed_kph :1.17
heading :107.97

GET https://api.thingspeak.com/update?api_key=4RX475FEXIOKAXHW&field1=-26.928816&field2=-49.064720&field3=1.17
```

Figura 19 – Execução do *firmware*.

Para a interação com o módulo SIM808 que está conectado ao Arduino será utilizado uma biblioteca fornecida pelo fabricante, a biblioteca oferece funções que facilitam a utilização do GPS e GPRS. Essa biblioteca nada mais é do que um conjunto de rotinas escritas em C/C++ que facilitam a programação do arduino que está conectado ao módulo SIM808.

4.3.1 Comunicação com módulo SIM 808

O módulo SIM808 está conectado à placa Arduino Uno e se comunica através de um canal serial, as conexões estão definidas no começo do código e foram escolhidos os pinos 10 para o RX e 11 para o TX conforme abordado na Seção 3.3.6. Foi desenvolvido um código para obter os dados do GPS e enviar com o GPRS estes dados para o *webservices* e repetir este procedimento até o microcontrolador ou o módulo serem desconectados,

isso foi feito utilizando os comandos da biblioteca própria do SIM808 e alguns comandos básicos do próprio Arduino.

Para atingir os objetivos escolhidos foi necessário dois elementos no desenvolvimento do código:

- Um canal serial lógico: Um canal serial lógico é iniciado com o objetivo de possibilitar a troca de informação entre algum componente e o arduino, neste caso, foi utilizado como canal de comunicação entre o Arduino Uno e o GPS do SIM808. Este canal foi iniciado através de uma biblioteca do arduino chamada SoftwareSerial.
- Comandos AT: Foi utilizado uma série de comandos AT para testar e fazer a comunicação entre o arduino e o GPRS e para configurar o cartão SIM.

4.3.2 Comunicação com webservice

A comunicação com o *webservice* é feita através da arquitetura REST e uma requisição HTTP GET, que envia os dados no padrão de estrutura JSON.

É recebido pelo *webservice* uma mensagem contendo 5 campos, cada um contendo um dado que será lido pelo *webservice*:

- created_at: Neste campo é enviado a data e horário do recebimento dos dados.
- entry_id: Este campo é responsável por identificar o ID que recebeu os dados enviados.
- field1: Neste campo é enviado o valor da Latitude atual do GPS.
- field2: Neste campo é enviado o valor da Longitude atual do GPS.
- field3: Neste campo é enviado o valor da velocidade atual do GPS.

A Figura 20 representa a forma que a mensagem de dados JSON é recebida pelo *webservice*.


```
{"created_at":"2021-04-
22T16:38:09Z","entry_id":297,"field1":"-26.927813","field2":"-49.064610","field3":"1.19"}
```


Figura 20 – Dados JSON recebido pelo *webservice*.

4.4 ThingSpeak

Dentre os *webservices* disponíveis, para o desenvolvimento deste projeto foi escolhido o ThingSpeak, esta plataforma foi escolhida por ser gratuita e suprir todas as necessidades para desenvolvimento do projeto proposto, além de ser fácil de utilizar e permitir a visualização dos dados de forma interativa, através de painéis.

ThingSpeak é uma plataforma de serviço e *Application Programming Interface* (API) para IoT que permite receber e enviar dados para dispositivos conectados à internet. Os dados podem ser armazenados e visualizados em tempo real através de painéis interativos. Para enviar e receber dados, o ThingSpeak disponibiliza dois métodos, através do protocolo MQTT ou da arquitetura REST, como citado na Seção 3.4.2, será utilizada a arquitetura REST para comunicação com o ThingSpeak. Esta plataforma também apresenta integração com os softwares Matlab e Twitter, então de forma geral essa ferramenta possibilita a criação de sistemas IoT de forma interativa e fácil, sem a necessidade de criar servidores para armazenamento de dados.

Essa ferramenta pode ser utilizada de forma totalmente gratuita com algumas limitações, os usuários que utilizam o serviço gratuito possuem limite de 15 segundos de intervalo para envio de dados e tem restrição de 3 milhões de envios por ano. Neste caso, mesmo deixando o sistema ligado continuamente com o envio de dados a cada 15 segundos, o consumo anual de 3 milhões de envios não seria atingido, então através do serviço gratuito do ThingSpeak seria possível manter o sistema em contínuo funcionamento. Só seria necessário obter uma fonte de dados de internet que suprisse o envio de dados gasto por um ano para manter o sistema em funcionamento contínuo.

A Figura 21 representa de forma ilustrativa o funcionamento do ThingSpeak.

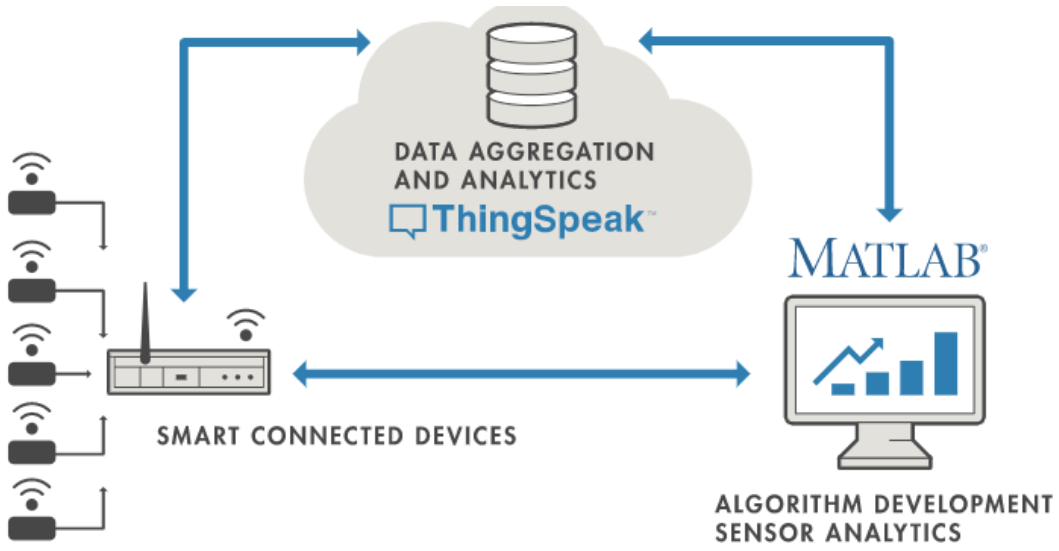


Figura 21 – Diagrama de comunicação de dispositivos com ThingSpeak [28].

4.4.1 Configuração do ThingSpeak

Para utilizar o ThingSpeak inicialmente foi necessário realizar um cadastro na plataforma e em seguida criar o canal que será utilizado pelo dispositivo do projeto, como é ilustrado na Figura 22.

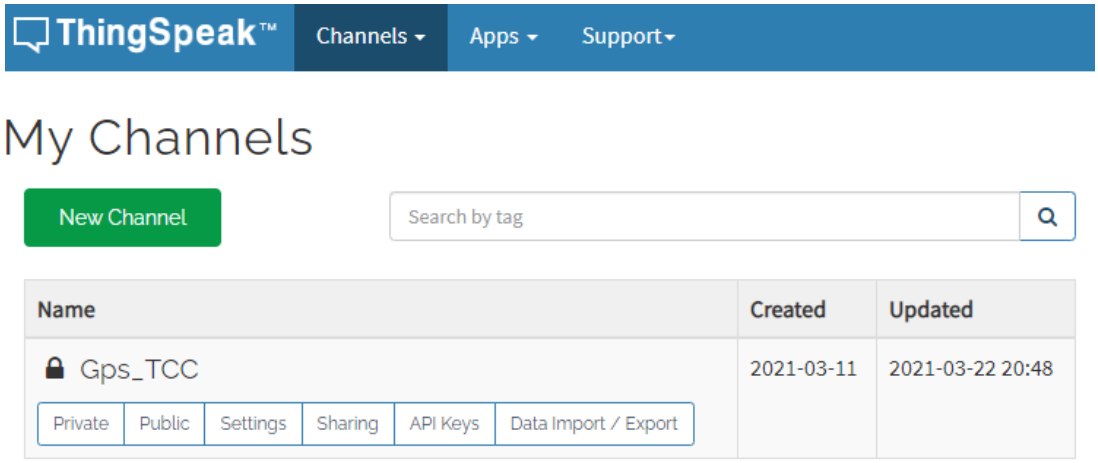


Figura 22 – Canal criado no ThingSpeak.

Após a criação do canal foi feita a configuração do mesmo para visualização dos dados que serão enviados. Para a configuração do canal, foram adicionados três campos: Longitude, Latitude e Velocidade. Os campos são enviados por GPRS através de uma requisição HTTP GET, como foi mencionado na Seção 3.5.

A Figura 23 mostra como foram feitas as configurações do canal. Cada um desses *fields* poderá ser visualizado na parte de *view* da plataforma.

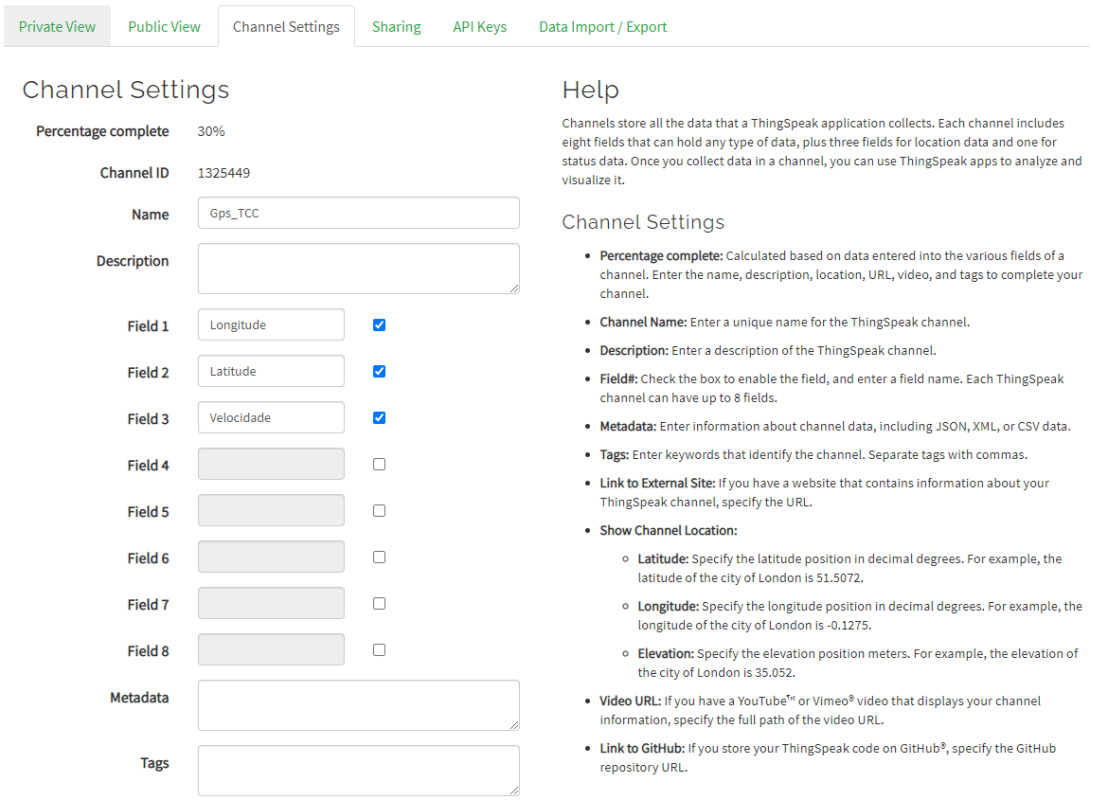


Figura 23 – Configuração do canal criado no ThingSpeak.

Após a configuração do canal, na parte de *view* da plataforma é possível visualizar um

painel interativo com os *fields* escolhidos, sempre que o ThingSpeak receber novos dados, estes dados serão adicionados a estes painéis gráficos. Estes gráficos são compostos por todo o histórico de dados enviados e suas respectivas datas, como mostra a Figura 24.

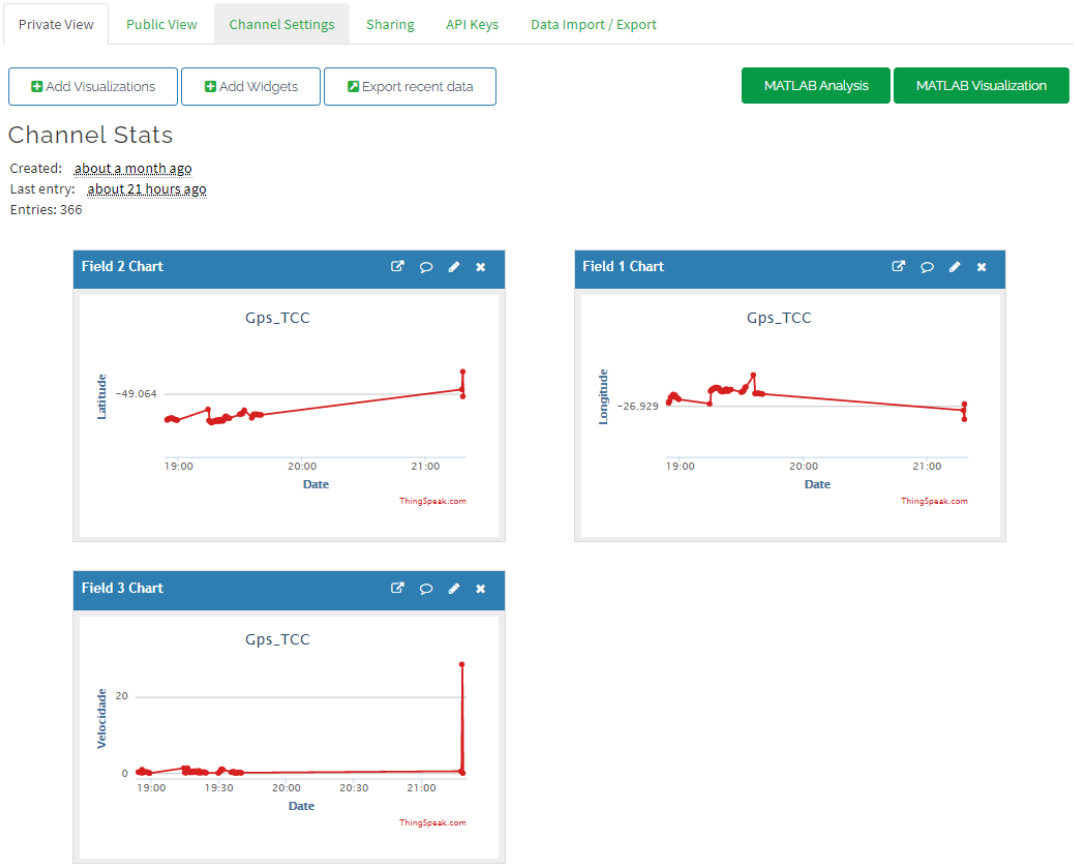


Figura 24 – Painéis de visualização de dados no ThingSpeak.

4.5 Aplicativo

Para melhor exposição dos dados obtidos, como foi citado em outras seções, será desenvolvido um aplicativo para sistemas Android que tem como principal objetivo mostrar de forma interativa a localização em tempo real do automóvel e do usuário, bem como algumas funcionalidades que melhoram a experiência do usuário ao utilizar o aplicativo. O desenvolvimento do aplicativo e as ferramentas utilizadas para sua construção serão mostradas ao longo desta seção.

4.5.1 Mit App Inventor

Para desenvolvimento do aplicativo proposto foi escolhido a plataforma MIT App Inventor, essa plataforma foi desenvolvida especificamente para criação de aplicativos e utiliza o método de programação em blocos, resolveu-se escolher esta plataforma por além de ser totalmente grátis, apresentar todas as ferramentas necessárias para o desenvolvimento do aplicativo proposto, utilizar uma forma de programação menos complexa e diferente das outras opções analisadas e possuir excelentes documentações de auxilio na Internet.

O MIT App Inventor é uma ferramenta desenvolvida pela Google, atualmente mantida pelo Instituto de Tecnologia de Massachusetts, que permite a criação de aplicativos para smartphones e tablets que possuem Sistema Operacional Android, a programação é realizada por blocos, onde cada bloco tem sua própria função e juntos podem formam uma lógica. Sendo então uma abordagem mais simplificada de programação mas que apresenta resultados satisfatórios, semelhantes a programação por código [29].

O App Inventor é uma plataforma de código aberto que tem como foco principal tornar, através de abordagens didáticas a programação e criação de aplicativos acessíveis para uma grande variedade de públicos, como: Educadores formais e informais, funcionários civis, pesquisadores, entusiastas e empreendedores, entre outros públicos [30].

A Figura 25 mostra a interface de *design* do MIT app inventor.

4.5.2 Programação em blocos

A metodologia de programação em blocos tem como foco a apresentação de uma forma simplificada e prática de programar, sendo utilizada para ensino de conceitos básicos e lógica de programação e desenvolvimento de *software*.

Os blocos são ferramentas de programação completamente diferentes dos métodos convencionais, neste caso, os blocos substituem as linhas de códigos escritas e cada bloco passa a ter um formato e cor específico que remete sua função, a combinação deste blocos forma o programa e então a aplicação desejada, exatamente da mesma forma que a programação por código, só que com algumas limitações.

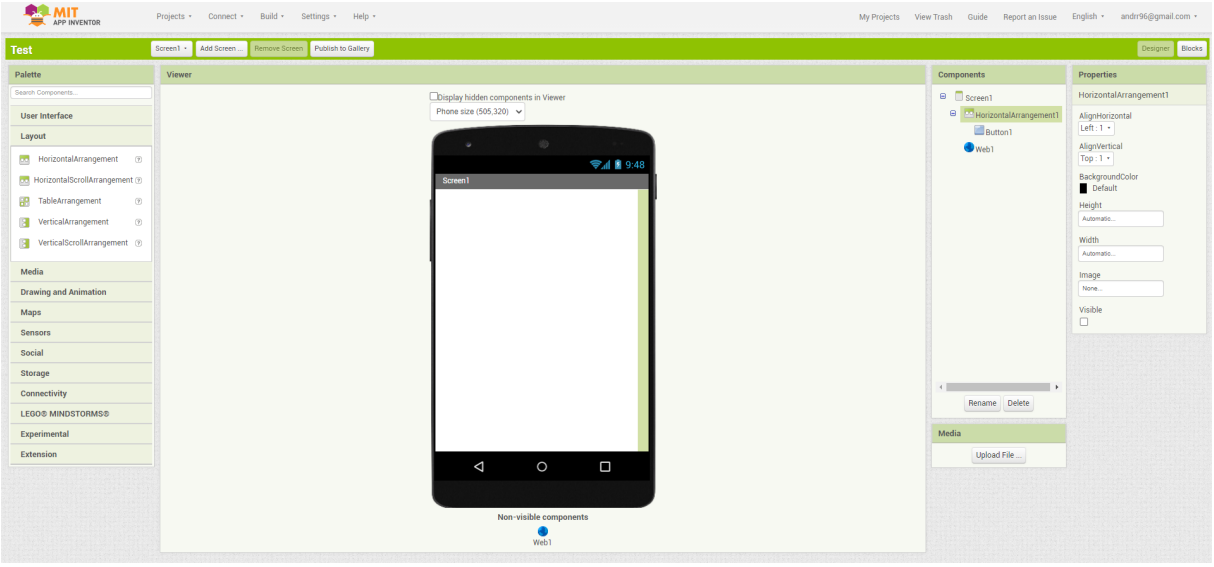


Figura 25 – Interface para programação do MIT App Inventor [31].

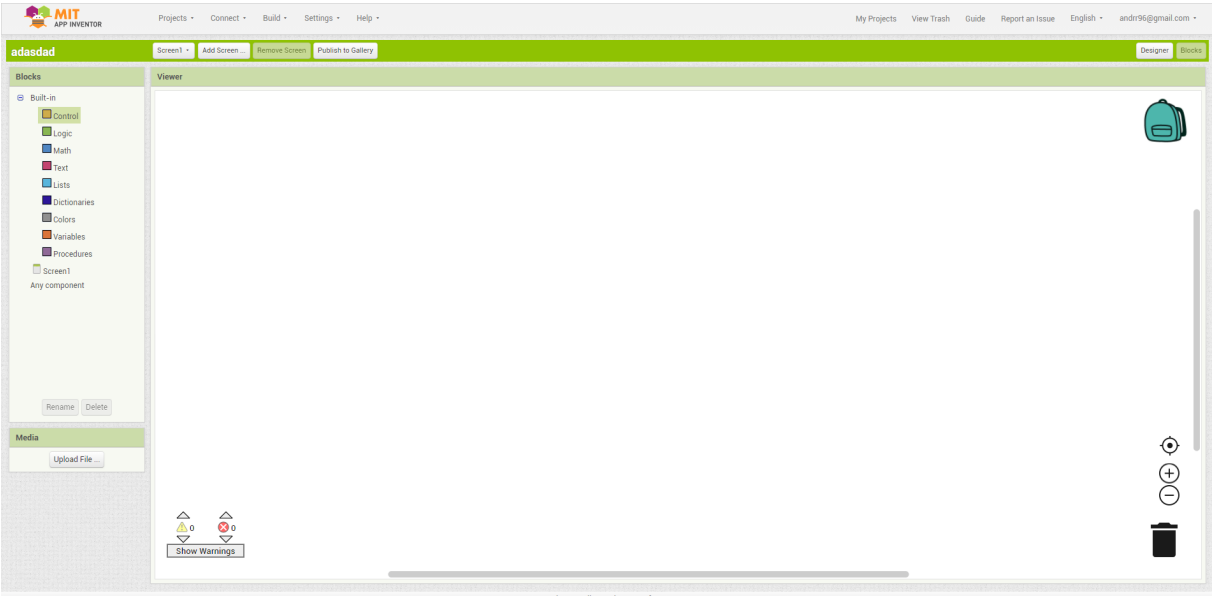


Figura 26 – Interface para programação em blocos do MIT App Inventor.

A Figura 26 mostra a interface de programação do MIT App inventor, lá será desenvolvida a programação em blocos do aplicativo proposto.

Como visto na Figura 26 os blocos de programação disponíveis apresentam as seguintes categorias: *Control*(Controle), *Logic*(Lógica), *Math*(Matemática), *Text*(Texto), *Lists*(Lista),*Colors*(Cores), *Variables*(Variáveis), *Procedures*(Procedimentos).

- *Control*: Este grupo de blocos de forma geral é responsável por realizar o controle das funcionalidades do aplicativo, como por exemplo, ao clicar em um botão fechar o aplicativo.

- *Logic*: Neste grupo contém os operadores lógicos como, verdadeiro ou falso, compa-

ração com igual, e, ou.

- *Math*: Bloco de operações matemáticas, contém funções básicas como adição, subtração, divisão e funções mais complexas, como trigonométricas.
- *Text*: Bloco de funções para se utilizar texto, como escrever algo, unir sentenças, comparar textos, entre outras.
- *Lists*: Estes blocos são utilizados de forma geral para manipulação de listas. É possível criar listas, fazer inserção e remoção de itens, fazer buscas de itens nas listas, entre outras funcionalidades.
- *Dictionaries*: Bloco utilizado para organização de dados, junto com o bloco *Lists* possibilitam a manipulação de bancos de dados.
- *Colors*: Grupo de blocos para edição de cores nos elementos do aplicativo.
- *Variables*: Este bloco apresenta a possibilidade criar variáveis globais, através de *get* e textitset é possível obter e declarar valores de variáveis.
- *Procedures*: Utilizado para criação de funções e procedimentos.

A partir do MIT App Inventor e da programação em blocos será desenvolvido então, o aplicativo utilizado neste projeto.

4.5.3 Desenvolvimento do aplicativo

Foi desenvolvido um aplicativo para sistemas Android, este aplicativo tem o objetivo de mostrar em sua interface os valores da latitude e longitude atual do automóvel, um mapa que apresenta a localização atual do automóvel e do dispositivo móvel que está utilizando o aplicativo, ou seja, toda vez que o usuário ou o veículo se movimentarem a localização será atualizada no mapa. O aplicativo também apresenta um botão que permite ao usuário obter uma rota otimizada no mapa entre a sua localização e a do automóvel, um botão para centralizar o mapa no automóvel e uma funcionalidade que permite ao usuário utilizar o Google Maps dentro do aplicativo.

O desenvolvimento do aplicativo foi separado em quatro partes:

- Inicialmente foi obtido e salvo em variáveis os valores que estão armazenados no *webservice* ThingSpeak.
- A segunda parte desenvolvida permite que o aplicativo receba a localização em tempo real do usuário e do automóvel no mapa.

- Em seguida, foi desenvolvido as funcionalidades, um botão que possibilita traçar rotas otimizadas entre o usuário e o automóvel, e a funcionalidade que permite ao usuário, através de um botão, visualizar a localização do automóvel no Google Maps, bem como traçar as rotas por lá também.

- Por último foi adicionado um botão para centralizar o mapa onde o usuário está localizado e também algumas melhorias, como sistemas de *zoom* e rotação do mapa e centralização do mapa com *double click*.

Em seguida será demonstrado e explicado como foram desenvolvidos, através de programação em blocos no MIT App Inventor, cada uma das quatro partes do aplicativo citadas acima.

4.5.4 Programação em blocos do aplicativo

Para desenvolver a primeira parte do aplicativo, foram utilizados alguns blocos de inicialização de variável, que foram utilizados ao longo do desenvolvimento de todo o aplicativo, e dois conjuntos de blocos como mostra Figura 27.

O primeiro conjunto de blocos é responsável por fazer uma solicitação do tipo HTTP GET que possibilita receber os dados de localização do GPS salvos no *webservice* ThingSpeak, essa solicitação é feita através de um link e uma *Key* retirados do canal do ThingSpeak, esses dados são recebidos em estrutura de texto do tipo JSON.

O segundo grupo de blocos tem a função de gravar os valores da latitude de longitude obtidos, esses valores são salvos em variáveis globais com seus respectivos nomes, essas variáveis poderão ser utilizadas ao longo do desenvolvimento do aplicativo em outros conjuntos de blocos que necessitem utilizar estes valores.

Na segunda parte do desenvolvimento do aplicativo foram utilizados dois conjuntos de blocos como mostra a Figura 28, o primeiro conjunto tem a função apenas de ativar o sensor de localização do usuário e atualizar no mapa a sua posição em tempo real. O segundo grupo de blocos é usado primeiramente para escrever a latitude e longitude do automóvel na tela inicial do aplicativo, para isso é necessário ler as variáveis em formato de texto, em seguida é obtido a localização do automóvel em tempo real no mapa através da variáveis globais latitude e longitude adquiridas na primeira parte do desenvolvimento do aplicativo.

A terceira parte do aplicativo foi desenvolvida com intuito de utilizar duas APIs, uma para gerar rotas otimizadas entre o usuário e automóvel, disponibilizada pelo *website* OpenRouteService, e a outra para utilizar os serviços padrão do Google Maps dentro do próprio aplicativo, disponibilizada pelo próprio Google Maps. Para que isso seja feito, são utilizados três grupos de blocos, como mostra a Figura 29.

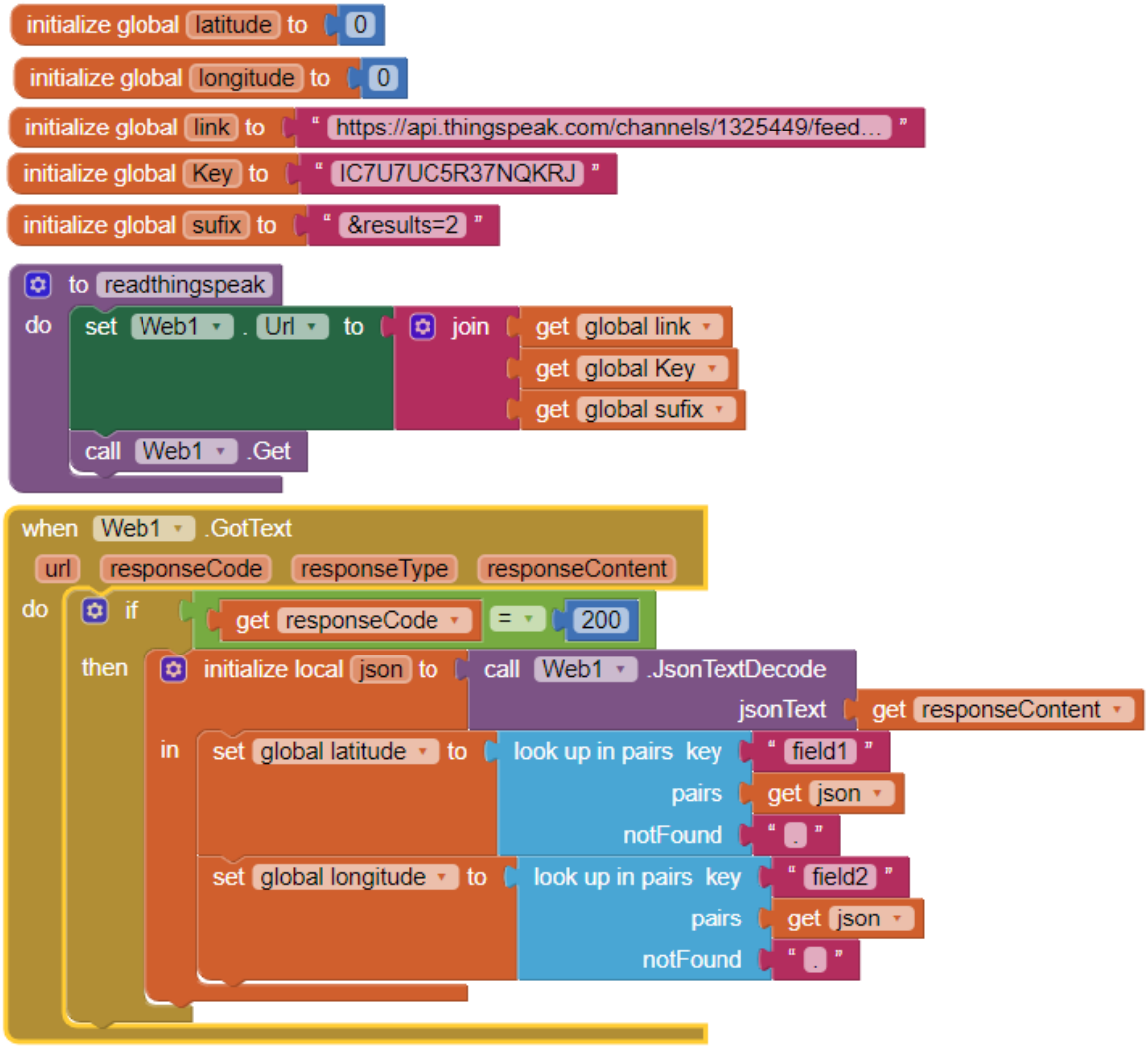


Figura 27 – Primeira parte da programação em blocos do aplicativo.

Inicialmente foi criado um botão chamado Traçar Rota, e ao acionar este botão é gerada uma rota otimizada através do serviço de rotas do *website* OpenRouteService, este *website* permite ao usuário obter de forma gratuita 3 mil rotas otimizadas diariamente. Para vincular ao MIT App Inventor é necessário usar apenas uma ferramenta do próprio App Inventor chamada *Navigation* (utilizado no primeiro conjunto de blocos da Figura 29), realizar um cadastro no OpenRouteService e através de uma *ApiKey* obter a rota otimizada.

O último conjunto de blocos desta parte do desenvolvimento é responsável por aplicar uma API do Google Maps que possibilita ao usuário manusear todas as funcionalidades presentes no serviço padrão do próprio Google Maps e já inserir automaticamente a localização atual do veículo. Para isso só foi necessário empregar as variáveis globais (latitude e longitude obtidas nas primeiras partes do desenvolvimento do aplicativo) junto com um bloco de ação e a forma de exibição padrão utilizada para iniciar o Google Maps através de API em sistemas Android.

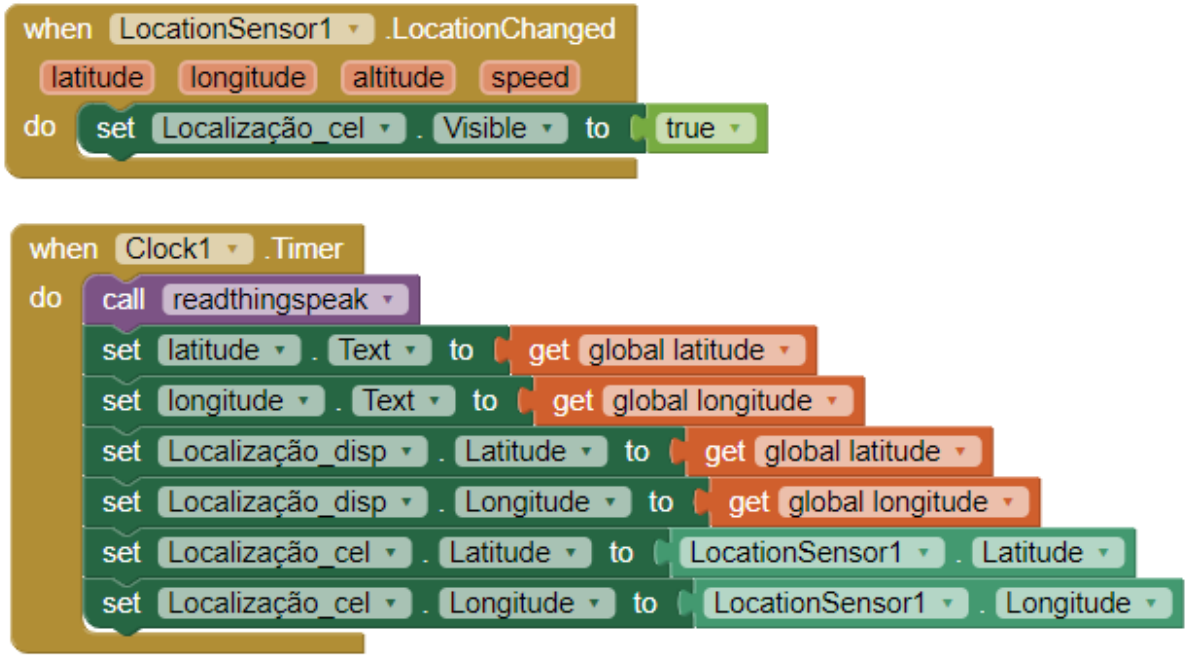


Figura 28 – Segunda parte da programação em blocos do aplicativo.

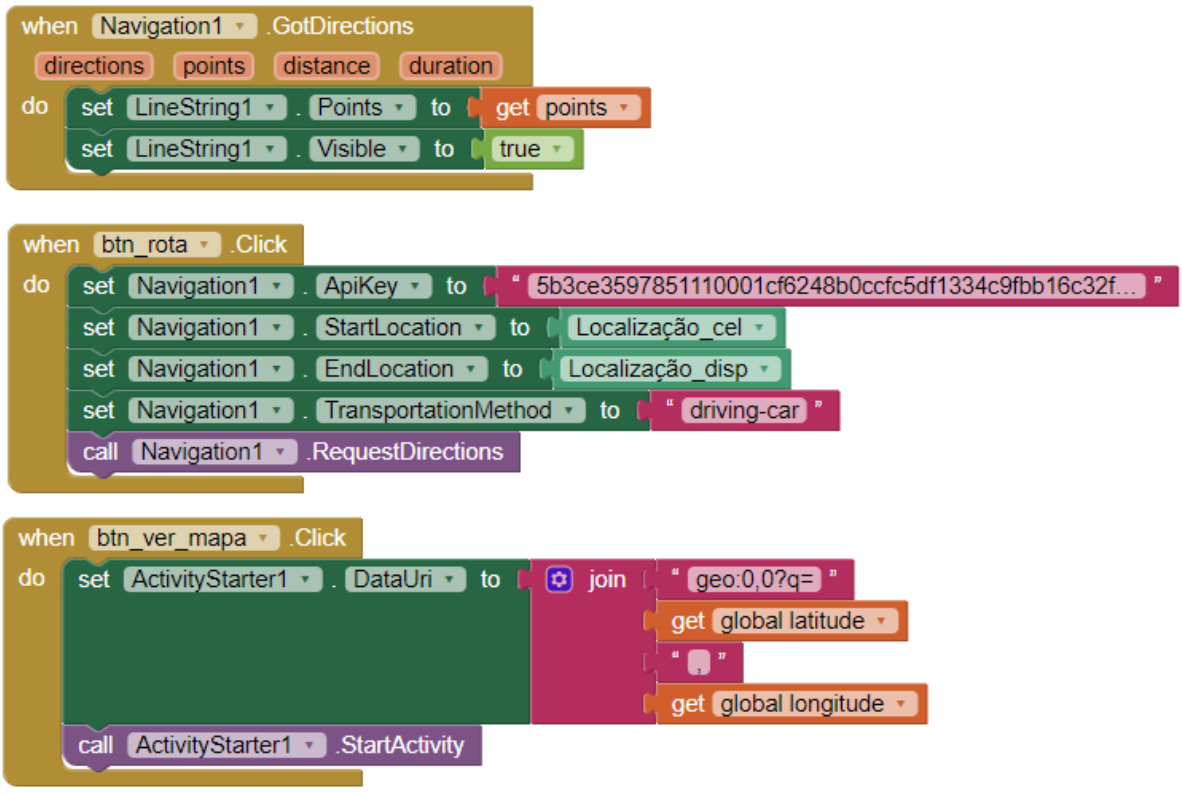


Figura 29 – Terceira parte da programação em blocos do aplicativo.

Por fim foi adicionado ao aplicativo um botão para centralizar o mapa na localização em que o usuário se encontra, e algumas melhorias básicas como sistemas de *zoom* e

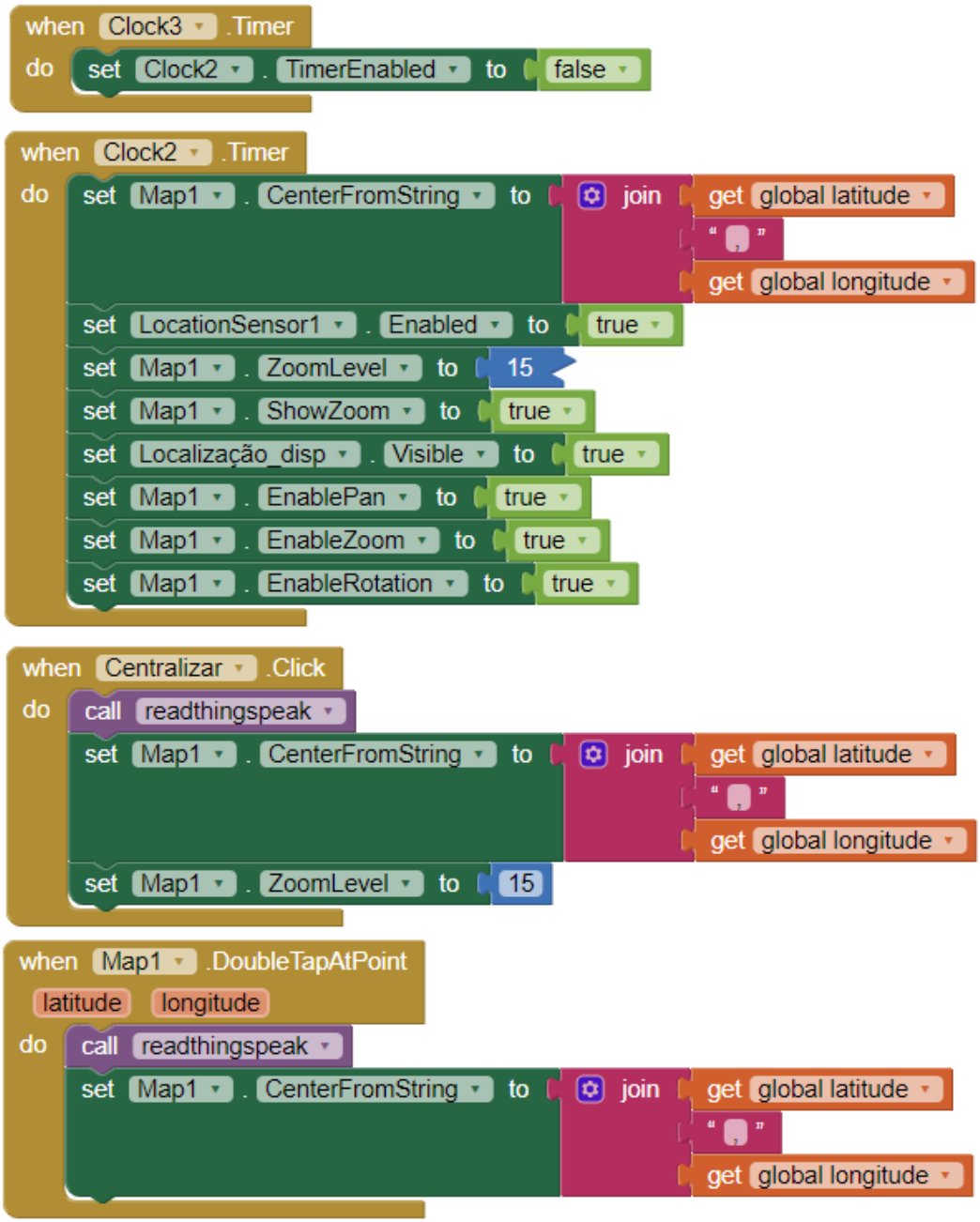


Figura 30 – Quarta parte da programação em blocos do aplicativo.

rotação do mapa e centralização do mapa com *double click*. A Figura 30 ilustra o diagrama de blocos programado para realizar essas tarefas.

5 Resultados

Neste capítulo serão apresentados os resultados obtidos através da parte final do sistema, ou seja o aplicativo, será mostrado a interface desenvolvida no MIT App Inventor e através de um dispositivo móvel Android será mostrado como funciona o aplicativo e suas principais funcionalidades disponibilizadas para o usuário.

5.1 Interface do aplicativo

A interface do aplicativo desenvolvido no MIT App Inventor está representada na Figura 31. Como mencionado nas outras seções, inicialmente na interface será mostrada a latitude e longitude do automóvel, um mapa com as localizações do usuário (símbolo vermelho) e do automóvel (símbolo verde) e algumas funcionalidades, um botão para centralizar o mapa, um botão para traçar a rota entre o usuário e o automóvel e um botão que permite ao usuário utilizar as funcionalidades do Google Maps dentro do aplicativo.

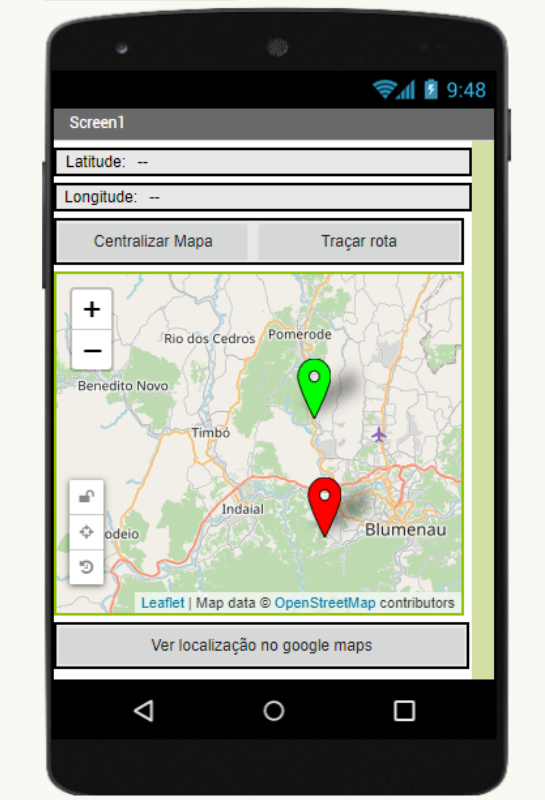


Figura 31 – Interface do aplicativo no MIT App Inventor.

5.2 Interface do aplicativo no dispositivo móvel

A Figura 32 representa o aplicativo desenvolvido em funcionamento, essa imagem foi gerada através de um dispositivo móvel Android com o sistema de rastreamento ligado, neste caso não foi acionado nem uma das funcionalidades, obteve-se então as coordenadas de latitude e longitude e as localizações do usuário (símbolo vermelho) e do veículo (símbolo verde) no mapa.

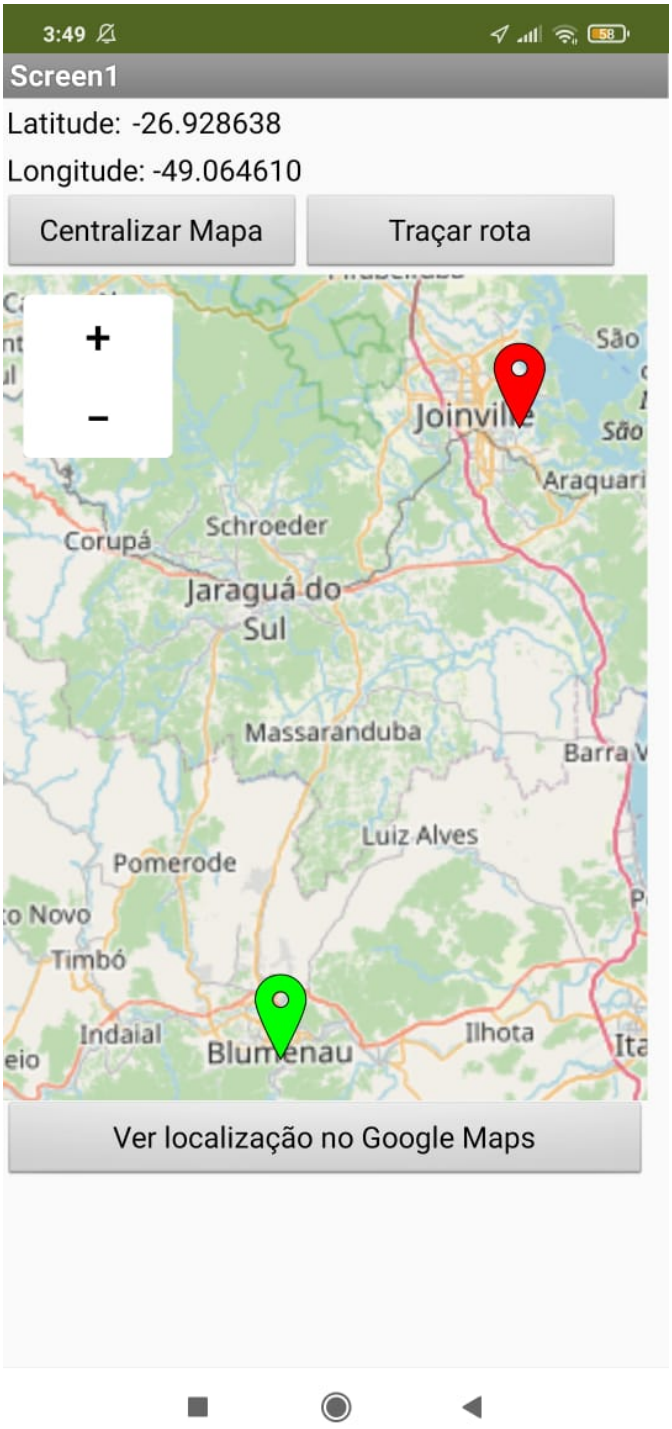


Figura 32 – Interface do aplicativo em um dispositivo móvel.

5.3 Geração de rotas

A Figura 33 representa novamente o aplicativo em uso em um dispositivo móvel Android, neste caso através do botão Traçar rota, presente no aplicativo, foi acionada a funcionalidade que permite ao usuário gerar uma rota otimizada entre a sua localização atual (simbolizado pelo ícone vermelho) e a localização do dispositivo acoplado ao veículo (simbolizado pelo ícone verde), a rota é representada pelo traço em cor preta.

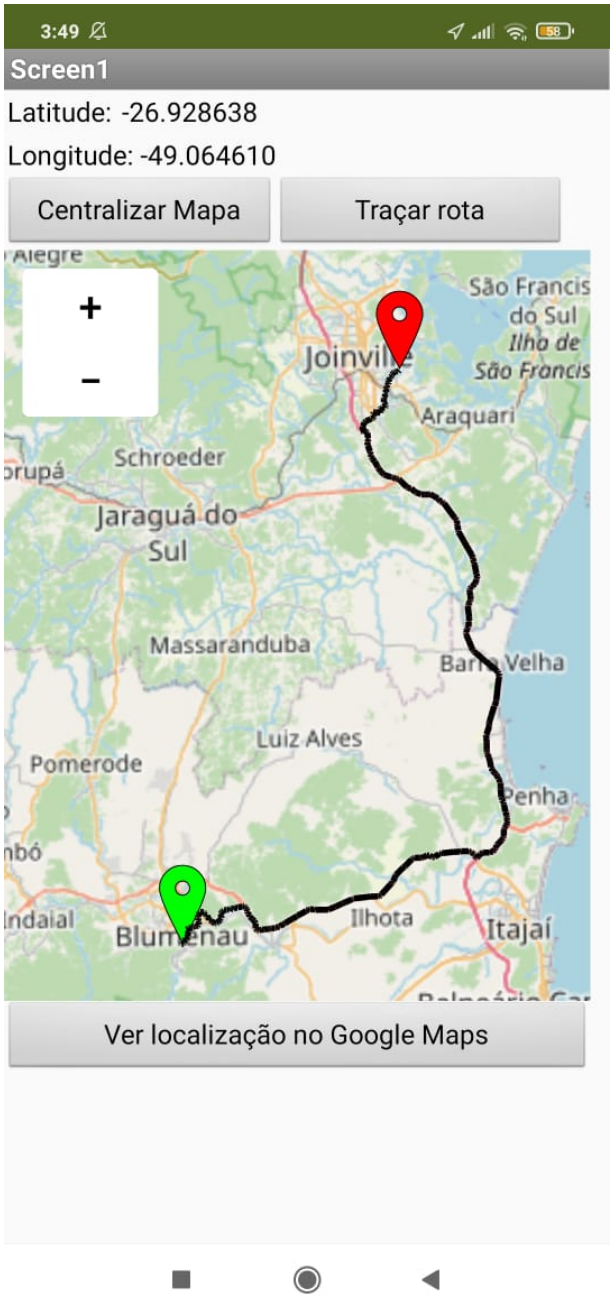


Figura 33 – Aplicativo com funcionalidade de geração de rotas.

5.4 Localização pelo Google Maps

A funcionalidade representada pela Figura 34 é acionada através do botão ver localização no Google Maps presente no aplicativo, esta funcionalidade através de uma API do google maps permite que o usuário utilize os serviços do google maps para obter a sua própria localização atual e a localização do veículo, bem como gerar uma rota entre as duas localizações, percebe-se que as duas funcionalidades geram a mesma rota, porém o Google Maps apresenta mais detalhes em sua interface.

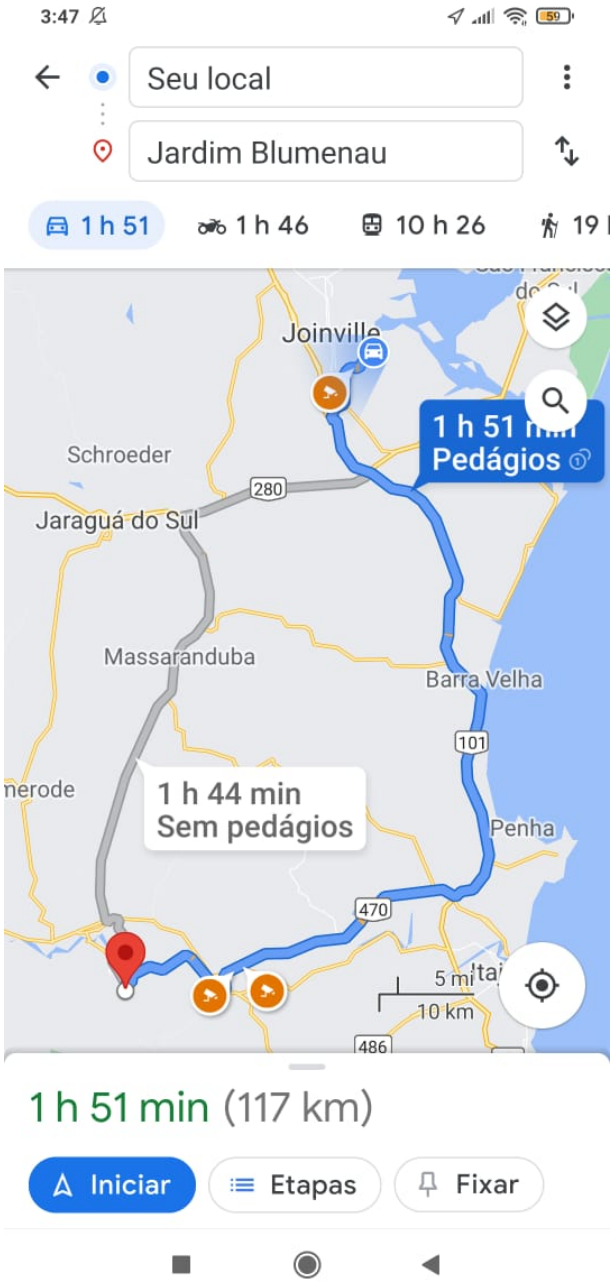


Figura 34 – Aplicativo com a funcionalidade ver localização no Google Maps.

5.5 Custos do projeto

A Tabela 3 apresenta os custos dos equipamentos utilizados para desenvolvimento do protótipo.

Tabela 3 – Tabela de custos.

Componente	Quantidade	Valor (R$)
Arduino Uno	1	42,90
Módulo SIM 808	1	97,78
Jumpers	10	5,00
Fonte 12V	1	11,00
Cartão SIM	1	30,00

O gasto final do protótipo foi de 186,68 reais.

6 Conclusões

Os crimes de furto e roubo de veículos vem crescendo de forma elevada nos últimos tempos conforme a tecnologia automotiva vem se tornando mais acessível para a população, com isso a preocupação com a segurança dos próprios bens tem se tornado cada vez maior. No caso de veículos a recuperação do bem é de extrema dificuldade, já que o proprietário não tem poder de reação e o tempo necessário para solicitar as autoridades muitas vezes acaba proporcionando o tempo suficiente para o assaltante fugir. Nesta situação é de extrema utilidade possuir um mecanismo ligado ao veículo que permite ao proprietário maiores chances de recuperar o veículo de forma que não ponha em risco a sua integridade e a de outros integrantes que o acompanham.

O presente projeto apresenta o desenvolvimento de um sistema para rastreamento veicular via GPS através da plataforma Arduino, ou seja é um sistema que fornece informações sobre o posicionamento do veículo em tempo real através de coordenadas geográficas. O sistema proposto faz uso de um microcontrolador Arduino Uno e um módulo conhecido como SIM808 que é composto por um módulo de GPS, que é utilizado para obter as coordenadas geográficas, e um módulo GSM/GPRS que tem a responsabilidade de enviar os dados obtidos a um servidor e através do servidor é possível enviar os dados para o aplicativo onde serão mostrados de forma mais interativa para os usuários.

O sistema final apresenta em um aplicativo os dados de localização do veículo em tempo real, com um sistema de rotas otimizadas para mostrar o melhor caminho entre o usuário e o veículo, possibilitando então uma ferramenta de extrema utilidade para localizar o automóvel e facilitar a recuperação do mesmo pelas autoridades, sem que seja necessário o envolvimento do proprietário.

Tendo em vista os estudos realizados e todo o sistema desenvolvido, composto pelo *hardware*, *software* e o aplicativo, nos resultados apresentados é possível concluir que os objetivos propostos para este projeto foram alcançados com sucesso.

6.1 Trabalhos futuros

Para fins de aprimoramento do sistema desenvolvido sugerem-se algumas propostas e questões importantes:

- Desenvolvimento de um sistema de alarme que pode ser integrado ao presente sistemas para alertar o usuário em caso de furto, para então melhores chances de recuperação do veículo furtado.

- Criar um sistema de alimentação próprio para manter o módulo embarcado funcionando sem depender da bateria do automóvel.
- Instalação de uma câmera no interior do veículo que pode ser integrada ao *webservice* para obtenção das imagens e então facilitar na identificação do criminoso.
- Melhorar o tempo de envio de dados entre os sistemas, atualmente devido a limitações do *webservice* o tempo de envio é de 15 segundos.
- Melhorias no aplicativo de forma que apresenta mais funcionalidades para tornar a experiência do usuário melhor e mais didática.

Referências Bibliográficas

1 GARRIDO, A. C. O. Fatores sociais de criminalidade. Minas Gerais, 2007. 13

2 SINESP. Estatísticas de roubo e furto entre 2014 e 2019, endereço: https://noticias.r7.com/sao-paulo/roubo-de-veiculos-ultrapassa-marca-de-1-milhao-no-brasil-em-4-anos-11102019. Acesso em 29/03/2021. 13

3 WOLF, W. Computer as components: Principles of embedded computing system. Embedded Computing System Design. 2nd Edition. Morgan Kaufmann Publishers, 2001. 15

4 LI Q.;YAO, C. Real-time concepts for embedded systems. 2003. 15

5 NOERGAARD, T. Embedded systems architecture: A comprehensive guide for engineers and programmers. Newnes, p. 656, 2005. 15

6 R, Z. Embedded systems. Taylor and Francis Group, LLC,New York, 2006. 15

7 KERSCHBAUMER, R. Microcontroladores. p. 10–11. 16

8 ARDUINO. Introdução arduino, endereço: https://www.arduino.cc/en/guide/introduction. Acesso em 29/03/2021. 17

9 ARDUINO. Arduino products, endereço: https://www.arduino.cc/en/main/products. Acesso em 29/03/2021. 17

10 MONICO, J. Posicionamento pelo gnss: Descrição, fundamentos e aplicações. Unesp, São Paulo, 2008. 19, 20

11 HOFFMANN, B. L. W.; COLLINS, J. Gps: Theory and practice. Springer-Verlag 3a ed, New York, 1994. 20

12 GUIMARAS, D. A. Introdução às comunicações móves, endereço: https://www.inatel.br/docentes/dayan/easyfolder/publications/3.pdf. Acesso em: 07/04/2021. 20

13 SOUZA, A. J. D. J. Controle e monitoramento residencial utilizando redes gprs. Universidade Estadual do Sudoeste da Bahia, Vitória da conquista, 2011. 21

14 CORDEIRO, G. R. Quarta geração de telefonia móvel. Universidade Federal do Paraná, Paraná, 2012. 23

15 TRAVIZANI, F. Proposta de um sistema de monitoramento em tempo real, para estudos de determinação de economia de energia, em aquecimento solar de Água. Universidade Estadual de Londrina, Londrina, 2018. 23

16 TAMAE, R. S. Sistema de processamento distribuído de imagens médicas com xml. Qualificação de Mestrado. PPGCC-UNIVEM – Fundação Eurípides de Marília, São Paulo, Março de 2004. 24

17 ARCHITECTURE, W. S. Relationship to the world wide web and rest architectures. World Wide Web Consortium, 11 de fevereiro de 2004. 25

18 TORRES ATSLANDS R. ROCHA, J. N. d. S. A. B. B. Analise de desempenho de brokers mqtt em sistema de baixo custo. XXXVI Congresso da Sociedade Brasileira de Computação, Julho 2016. 26

19 BESZCZYNSKI, L. ProtÓtipo de um sistema de rastreamento veicular baseado no modulo telit. Universidade Regional de Blumenau, Blumenau, 2008. 27

20 AL., L. W. et. Implementação do sistema de mapeamento de uma linha de Ônibus para um sistema de transporte inteligente. Departamento de Ciência da Computação – Universidade de Brasília (UnB), Janeiro, 2001. 28

21 SIMIAO, R. M. Rastreador veicular via web. Assis, 2009. 29

22 SHOKEEN, S. The importance of gps tracking system, endereço: https://www.trackyoursawari.com/blog/the-importance-of-gps-tracking-system/. Acesso em: 15/04/2021. 31

23 ARDUINO. Especificações arduino uno, endereço: https://store.arduino.cc/usa/arduino-uno-rev3. Acesso em 30/03/2021. 32, 33

24 WANZELER, T. M. AutomaÇÃo residencial de baixo custo utilizando a plataforma de prototipagem eletrÔnica arduino. Tucurui, 2015. 33

25 LOBODAROBOTICA. Pinagem atmega328, endereço: https://lobodarobotica.com/blog/arduino-uno-pinout/. Acesso em 01/04/2021. 34

26 MILLER, J. Build a car tracking system with the sim808 module, endereço: https://maker.pro/arduino/projects/build-a-car-tracking-system-with-the-sim808-module. Acesso em: 20/04/2021. 35, 37

27 ITEAD. Sim808 gsm/gprs/gps module, endereço: https://www.itead.cc/wiki/sim808_gsm/gprs/gps_module. Acesso em: 20/04/2021. 36

28 THINGSPEAK. Learn more about thingspeak, endereço: https://thingspeak.com/pages/learn_more. Acesso em: 23/04/2021. 42

29 SILVA IGOR GALINDO PAIVA, D. X. F. Aricio Medeiros da. Desenvolvimento de aplicativo para android com uso do mit app inventor. Revista Científica da FASETE 2017.2, p. 191–203, Acesso em: 25/04/2021. 45

30 MIT APP INVENTOR, Endereço: https://aedmoodle.ufpa.br/ plugin-file.php/170097/mod_book/chapter/2317/MITAPP_Inventor_apostila.pdf. Acesso em: 25/04/2021. 45

31 ALBRITTON, R. What is app inventor?, endereço:https://learn.adafruit.com/mit-app-inventor-and-particle-io. Acesso em: 25/04/2021. 46