

Fábio Verrone

## **SISTEMA DE RASTREAMENTO E BLOQUEIO DE VEÍCULOS**

Orientador: Prof. Dr. Marco Túlio Carvalho de Andrade

**São Paulo**  
**2015**

Fábio Verrone

## **SISTEMA DE RASTREAMENTO E BLOQUEIO DE VEÍCULOS**

Trabalho de Conclusão de Curso – Escola Politécnica  
da Universidade de São Paulo – Engenharia Elétrica  
com ênfase em Computação.

Orientador: Professor Doutor  
Marco Túlio Carvalho de Andrade

**São Paulo**  
**2015**

## **AGRADECIMENTOS**

À minha família e amigos, por todo incentivo e paciência, essenciais à minha formação acadêmica.

Aos professores da Escola Politécnica da Universidade de São Paulo, por me fornecerem o conhecimento necessário à elaboração e desenvolvimento deste projeto, em especial, meu orientador, Marco Túlio Carvalho de Andrade, por acreditar neste trabalho e por toda atenção, apoio e dedicação.

A todos que colaboraram de alguma forma neste projeto.

## RESUMO

Neste projeto foi desenvolvido um sistema de rastreamento e bloqueio de veículos, utilizando as tecnologias GPS, para obtenção da posição do veículo, e GSM para comunicação do sistema dentro do veículo com o telefone celular do usuário.

Foi utilizada uma plataforma de prototipagem eletrônica Arduino UNO, um módulo SIM908 que integra as tecnologias GSM e GPS, um cartão SIM e um computador para simular os sinais enviados e recebidos pelo veículo.

Foi desenvolvida uma interface gráfica para sistemas operacionais Android para facilitar a comunicação do usuário com o hardware embarcado e para permitir que o usuário veja a posição do veículo em um mapa.

O hardware dentro do veículo possui dois modos de operação: Ativo e Passivo. O modo Passivo somente responde às requisições feitas pelo usuário, enquanto que no modo Ativo uma mensagem SMS é enviada aos administradores e ao usuário indicado por eles, como motorista atual, caso o sistema de ignição do veículo, que se encontra desligado, seja acionado.

Até dois administradores podem ser cadastrados no cartão SIM. Se o cartão SIM estiver vazio, ao primeiro usuário a se cadastrar é permitido privilégios de administrador, e somente ele poderá cadastrar um novo usuário ou retirar um usuário do sistema.

Palavras Chave: GPS, GSM, Sistema Embarcado, Arduino, Android, SIM908.

## ABSTRACT

In this project was developed a vehicle tracking and blocking system, using GPS technology to obtain the vehicle position and GSM for communicating with the system inside the vehicle through the user's cell phone.

It was used an Arduino UNO electronic prototyping platform, a SIM908 module that integrates GSM and GPS technologies, a SIM card and a computer to simulate the signals sent and received by the vehicle.

A graphic Interface was developed to Android operating systems to facilitate the communication between the user and the embedded hardware and to permit the user to visualize the vehicle position in a map.

The hardware inside the vehicle possesses two operation modes: Active and Passive. The Passive mode only answers the requisitions made by the user, while in the Active mode a SMS message is sent to the administrators, and the user they indicate as the current driver, in case the ignition system of the vehicle, which is stopped, is turned on.

Up to two administrators can be registered in the SIM card. If the SIM card is empty, to the first user to register is given administrator privileges, and only the administrator can register a new user or remove a user from the system.

Keywords: GPS, GSM, Embedded hardware, Arduino, Android, SIM908

## LISTA DE ILUSTRAÇÕES

<b>Figura 1</b> - Constelação de Satélites.....	13
<b>Figura 2</b> - Estrutura de uma sentença GGA.....	16
<b>Figura 3</b> - Arquitetura GSM.....	18
<b>Figura 4</b> - Arquitetura SMS.....	18
<b>Figura 5</b> - Arduino UNO.....	19
<b>Figura 6</b> - Módulo SIM908.....	26
<b>Figura 7</b> - Diagrama Funcional do SIM908.....	28
<b>Figura 8</b> - Pinagem do módulo SIM908.....	28
<b>Figura 9</b> - Conexão entre Arduino e SIM908.....	29
<b>Figura 10</b> - Interface da antena GSM com o SIM908.....	30
<b>Figura 11</b> - Interface da antena GPS com o SIM908.....	33
<b>Figura 12</b> - Leitor Cartão SIM de 6 Pinos.....	34
<b>Figura 13</b> - Conexão do Leitor de Cartão SIM ao módulo SIM908.....	34
<b>Figura 14</b> - Conexão do <i>Shield</i> com o Arduino Uno.....	36
<b>Figura 15</b> - Integração do Hardware.....	36
<b>Figura 16</b> - Interface Android.....	38
<b>Figura 17</b> - Classe SMSSender.....	38
<b>Figura 18</b> - IDE Arduino.....	40
<b>Figura 19</b> - Resultado da função getLatLong().....	49
<b>Figura 20</b> - Estrutura da Mensagem SMS.....	50
<b>Figura 21</b> - Resultado da função enviaSms().....	51
<b>Figura 22</b> - Resposta do SIM908 ao SMS Enviado.....	52
<b>Figura 23</b> - Fluxograma de Mensagens.....	55
<b>Figura 24:</b> Estrutura do CD.....	56
<b>Figura 25</b> - Diagrama Arduino Uno.....	57

## LISTA DE QUADROS

<b>Quadro 1</b> - Formatos padrões relacionados ao GPS (GNSS).....	15
<b>Quadro 2</b> - Sentenças NMEA relacionadas ao GNSS.....	15
<b>Quadro 3</b> - Termos da Sentença GCA.....	16
<b>Quadro 4</b> - Especificações Técnicas do Arduino Uno.....	27
<b>Quadro 5</b> - Frequências de Operação do SIM908.....	30
<b>Quadro 6</b> - Modos de Operação do SIM908.....	31
<b>Quadro 7</b> - Comandos AT do GSM.....	32
<b>Quadro 8</b> - Comandos AT do GPS.....	33
<b>Quadro 9</b> - Pinos do módulo sim908 conectados ao Leitor de Cartão SIM.....	35
<b>Quadro 10</b> - Níveis máximos absolutos.....	58
<b>Quadro 11</b> - Condições Recomendadas de Operação.....	58

## LISTA DE ABREVIATURAS E SIGLAS

<b>ADT</b>	<i>Android Development Kit</i>
<b>API</b>	<i>Interface de Programação de Aplicativos</i>
<b>CB</b>	<i>Cell Broadcast</i>
<b>CEPT</b>	Conferência Europeia das Administrações de Correios e Telecomunicações
<b>CTS</b>	<i>Clear to Send</i>
<b>DCS</b>	<i>Digital Cellular Service</i>
<b>EGSM</b>	<i>Enhanced GSM</i>
<b>ETSI</b>	Instituto Europeu de Normas de Telecomunicações
<b>GLONASS</b>	Sistema de Navegação Global por Satélite
<b>GNSS</b>	Sistema de Navegação Global por Satélite
<b>GPRS</b>	Serviços Gerais de Pacote por Rádio
<b>GPS</b>	Sistema de Posicionamento Global
<b>GSM</b>	Sistema Global para Comunicação Móvel
<b>HRL</b>	Home Location Register
<b>IDE</b>	Ambiente de Desenvolvimento Integrado
<b>JDK</b>	<i>Java Development Kit</i>
<b>MO</b>	<i>Mobile Originated</i>
<b>MT</b>	<i>Mobile Terminated</i>
<b>PCS</b>	<i>Personal Communication System</i>
<b>RTK</b>	Levantamento Cinemático em Tempo Real
<b>RTS</b>	<i>Request to Send</i>
<b>SDK</b>	<i>Software Development Kit</i>
<b>SIM</b>	Módulo de identificação do Assinante
<b>SMS</b>	Short Message Service
<b>VRL</b>	Visitor Location Register



## SUMÁRIO

<b>1. INTRODUÇÃO.....</b>	<b>11</b>
1.1 OBJETIVO.....	11
1.2 METODOLOGIA.....	12
<b>2. REVISÃO BIBLIOGRÁFICA.....</b>	<b>13</b>
2.1 SISTEMA DE POSICIONAMENTO GLOBAL (GPS) .....	13
2.2 SISTEMA GLOBAL PARA COMUNICAÇÃO MÓVEL (GSM) .....	17
2.3 <i>SHORT MESSAGE SERVICE</i> (SMS) .....	18
2.4 SOLUÇÕES EXISTENTES.....	20
<b>3. DESCRIÇÃO DO PROBLEMA.....</b>	<b>22</b>
3.1 MODELO DO PROBLEMA.....	22
<b>3.1.1 Requisitos Funcionais Básicos.....</b>	<b>23</b>
<b>3.1.2 Requisitos Funcionais de Cadastro.....</b>	<b>23</b>
<b>3.1.3 Requisitos Funcionais de Mensagens.....</b>	<b>24</b>
<b>3.1.4 Requisitos Funcionais da Interface Android.....</b>	<b>24</b>
<b>3.1.5 Requisitos Não Funcionais.....</b>	<b>24</b>
<b>4. MODELO DA SOLUÇÃO.....</b>	<b>25</b>
4.1 HARDWARE.....	25
<b>4.1.1 Arduino Uno.....</b>	<b>26</b>
<b>4.1.2 Módulo SIM908.....</b>	<b>27</b>
<b>4.1.3 Leitor do Cartão SIM.....</b>	<b>34</b>
<b>4.1.4 Integração do Hardware.....</b>	<b>35</b>
4.2 SOFTWARE.....	37
<b>4.2.1 Interface Gráfica.....</b>	<b>37</b>
<b>4.2.2 IDE Arduino.....</b>	<b>39</b>
<b>5. ESTUDO DE CASO.....</b>	<b>41</b>
5.1 CADASTRO DO ADMINISTRADOR COM O CARTÃO SIM VAZIO....	41
5.2 CADASTRO DE USUÁRIO ADICIONAL E ATRIBUIÇÃO DE PRIVILÉGIOS.....	41

5.3 RETIRADA DE USUÁRIO CADASTRADO.....	42
5.4 PEDIDO DA POSIÇÃO DO VEÍCULO PELO USUÁRIO.....	42
5.5 PEDIDO DE BLOQUEIO DO VEÍCULO PELO USUÁRIO.....	42
5.6 MODO ATIVO E MODO PASSIVO.....	43
5.7 MENSAGENS DE REPOSTA DO HARDWARE EMBARCADO.....	43
<b>6. CONSIDERAÇÕES FINAIS.....</b>	<b>44</b>
6.1 CUMPRIMENTO DOS OBJETIVOS.....	44
6.2 CONTRIBUIÇÕES DO TRABALHO.....	45
6.3 TRABALHOS FUTUROS.....	45
6.4 CONCLUSÕES.....	46
<b>7. REFERÊNCIAS.....</b>	<b>47</b>
<b>APÊNDICE A – Testes com o SIM908.....</b>	<b>49</b>
<b>APÊNDICE B – Arquivo Manifest.xml.....</b>	<b>53</b>
<b>APÊNDICE C – Classe SMSReceiver.....</b>	<b>54</b>
<b>APÊNDICE D – Fluxograma de Mensagens.....</b>	<b>55</b>
<b>APÊNDICE E – Estrutura do CD.....</b>	<b>56</b>
<b>ANEXO A – Diagrama Arduino Uno.....</b>	<b>57</b>
<b>ANEXO B – Características Elétricas do SIM908.....</b>	<b>58</b>

# 1. INTRODUÇÃO

Dados da Secretaria de Segurança Pública do estado de São Paulo [10] mostram que as taxas de furtos e roubos de veículos, ainda que venha diminuindo nos últimos dez anos, é motivo de preocupação para as autoridades de segurança.

Através de dados coletados pelos departamentos da Polícia Civil, Polícia Militar e da Superintendência da Polícia Técnico-Científica, e divulgados pela SSP/SP [11], até julho de 2015 já haviam sido registradas 44.773 ocorrências de roubos e 65.452 furtos de veículos, sendo que destes, 52.713 veículos foram recuperados [10]. Também durante este período foram registradas 5.014 ocorrências de roubo de cargas, contra 5.033 do mesmo período em 2014 e 4.418 em 2013.

Deve ser considerado também o risco de vida a qual o motorista está sujeito em uma situação de roubo, principalmente quando há reação contra o criminoso, gerada pela incerteza do motorista quanto à recuperação de seu veículo.

A solução adotada para resolver o problema deve tentar minimizar a reação do motorista ao possibilitar o rastreamento do veículo e seu bloqueio para recuperação, assim como coibir ações futuras de criminosos.

## 1.1 OBJETIVO

O objetivo deste projeto é criar um sistema que seja capaz de monitorar o veículo de um usuário, e avisá-lo, através de mensagem de texto para seu telefone celular, quando o veículo não estiver mais parado. O sistema deve ser capaz de obter e enviar a posição global do veículo, e gerar sua interrupção imediata quando solicitado pelo usuário, através de mensagem de texto ao sistema, de seu telefone celular.

## 1.2 METODOLOGIA

Foram realizados estudos sobre os Ambientes de Desenvolvimento Integrado e seus complementos, necessários para a codificação do hardware dentro do veículo e da interface do dispositivo Android, especificações dos módulos utilizados no hardware embarcado e soluções já existentes para a resolução do problema descrito.

A interface de comunicação entre o telefone celular e o hardware dentro do veículo foi desenvolvida, e testes iniciais foram realizados para verificar o funcionamento de cada componente do hardware embarcado separadamente.

Após os testes iniciais os componentes do hardware dentro do veículo foram integrados, e o sistema de cadastros de usuários foi codificado, possibilitando a comunicação entre o hardware embarcado e o telefone celular.

Novos testes foram realizados, para verificar o envio de mensagens pré-definidas através da interface do telefone celular ao hardware dentro do veículo. O envio de mensagem SMS pelo módulo GSM, com as coordenadas obtidas pelo módulo GPS, ao telefone celular pelo hardware embarcado também foi verificado, assim como sua adequada visualização na interface.

Foram codificados o sistema de privilégios no hardware embarcado, e o simulador dos sistemas de ignição e bombeamento de combustível do veículo no computador conectado a sua porta serial.

Após o desenvolvimento do simulador dos sistemas presentes no veículo, foi possível codificar os modos Ativo e Passivo e realizar os testes finais, para verificar as modificações realizadas e a total integração do sistema.

### **Comentários Bibliográficos**

O estudo das IDEs e Complementos foi feito seguindo o roteiro proposto por DEITEL [6], e posteriormente utilizada na seção 4.2 deste documento.

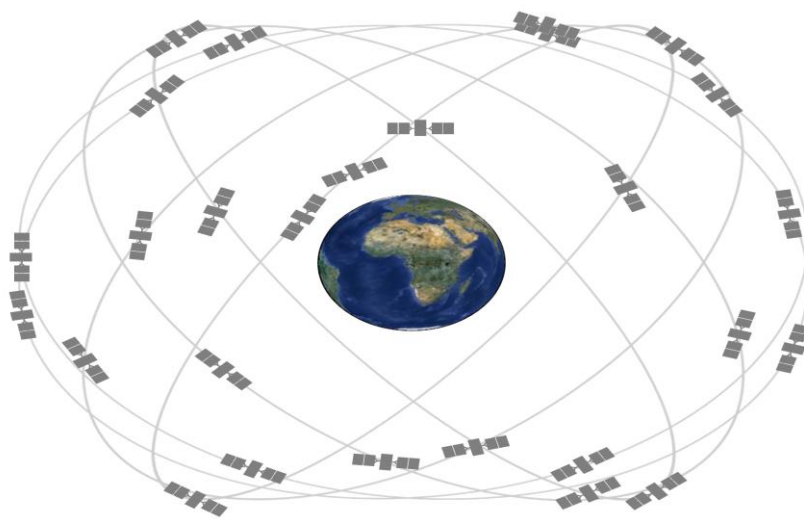
O estudo das especificações de hardware foi feito através dos manuais disponibilizados pela Shangai SIMCom wireless solution Ltd. [8] [9] e utilizado na seção 4.1 deste documento.

## 2. REVISÃO BIBLIOGRÁFICA

Neste capítulo são descritos os conceitos fundamentais nos quais a solução adotada se baseia, através de uma revisão de livros da bibliografia que se dedicam ao tema. Também é feita uma análise de soluções existentes destinadas a resolver o problema proposto, suas semelhanças e divergências com este projeto.

### 2.1 SISTEMA DE POSICIONAMENTO GLOBAL (GPS)

Conforme KAPLAR e HEGARTY (2006. p.3), GPS atende atualmente os critérios estabelecidos por diversas organizações governamentais dos Estados Unidos da América na década de 60, entre elas o Departamento de Defesa, a NASA e o Departamento de Transportes, fornecendo de forma precisa, contínua, em todo o mundo, a posição tridimensional e informação sobre a velocidade a usuários que possuam o equipamento de recepção adequado. O sistema consiste de 24 satélites arranjados em seis planos orbitais com quatro satélites por plano e uma rede de monitoramento distribuída por todo mundo.



**Figura 1** - Constelação de Satélites

**Fonte:** <http://www.gps.gov/multimedia/images/constellation.jpg>

Os satélites transmitiam códigos e dados de navegação originalmente em duas frequências de uso civil: L1 C/A(1575,42MHz), L2(1227,6MHz), porém como parte do programa de modernização do GPS [2], os satélites passarão a transmitir em quatro frequências de uso civil: L1 C/A(1575,42MHz), L2C(1227MHz), para uso comercial, que começou a ser transmitida em 2005, está presente em 17 satélites atualmente e prevista para os 24 satélites em 2018, L5(1176 MHz), para uso em aplicações de alto desempenho que começou a ser transmitida em 2010, está presente em 10 satélites atualmente e prevista para os 24 satélites em 2021 e L1C (1555 MHz), que possui previsão de lançamento com o GPS III em 2017 e para os 24 satélites até 2020. Cada satélite transmite nestas frequências, através de uma técnica chamada de acesso múltiplo por divisão de código(CDMA), porém utilizando códigos de aquisição(C/A) e precisão, P(Y), diferentes entre eles.

A técnica utilizada na obtenção da posição tridimensional do usuário usa o conceito de tempo de chegada do sinal (TOA). Os dados de navegação permitem que o receptor determine a localização do satélite no momento em que o sinal foi transmitido, enquanto os códigos C/A e P(Y) permitem que o receptor determine o tempo de propagação do sinal e, desta forma, a distância entre o satélite e o usuário. Essa técnica necessita que o receptor também possua um relógio, que se sincronizado com o relógio dos satélites, exige que a medição seja feita por três satélites, caso contrário, as medições devem ser feitas por quatro satélites.

Conforme EL-RABBANY(2006, p.103), grupos de pesquisa e agências criaram formatos padrões para diferentes necessidades de uso de dispositivos GPS. Uma breve descrição dos formatos mais utilizados, e as agências que os desenvolveram, é fornecida no Quadro 1.

Formato	Descrição	Agência
RINEX	Observações GPS e GLONASS, dados meteorológicos e dados de navegação.	IGS
BINEX	Formato de troca binária para os dados de GPS, GLONASS e SBA - principalmente para propósitos de pesquisa.	UNAVCO
SP3-c	Soluções de órbita para GPS e GLONASS.	IGS
SINEX	Soluções para a posição inicial e velocidade.	IGS
IONEX	Produtos de grade ionosférica TEC.	IGS

(Continuação)

Formato	Descrição	Agência
Tropo SINEX	Produtos para atraso zenital troposférico.	IGS
ANTEX	Calibração de Antenas.	IGS
RTCM SC-104	Correções e dados GNSS diferenciais e cinemáticos.	RTCM
RTCA SC-159	Correções diferenciais e seus limites de erro.	RTCA
NMEA 0183	Solução de posicionamento GNSS, tempo, e outras informações relacionadas ao GNSS e outros sensores.	NMEA

**Quadro 1** - Formatos padrões relacionados ao GPS (GNSS)  
**Fonte:** Tradução do Quadro 8.1 de EL-RABBANY(2006, p.103)

Segundo EL-RABBANY(2006, p.118), NMEA 0183 foi originalmente adotado como um formato para interligação de dispositivos eletrônicos marinhos, através de transmissão de dados em formato ASCII, transmitidos a uma taxa de 4800 bps entre dois dispositivos. Os dados são enviados em sentenças começando com o caractere “\$” seguido de um campo de endereço de cinco caracteres, que identifica o emissor (dois primeiros caracteres) e o formato e tipo de dados (três caracteres finais). Os campos de dados são separados por vírgulas. Um campo nulo (i.e., campo com comprimento zero) é utilizado quando o valor do campo não está disponível ou não é confiável. O último campo segue um caractere asterico e corresponde a um campo de soma de verificação (*checksum*). Todas as sentenças terminam com retorno de carro “<CR><LF>”. O número total de caracteres em qualquer sentença é 82 ( i.e., máximo de 79 caracteres entre “\$” e “<CR><LF>”)

Sentença NMEA	Descrição
ALM	Dados de almanaque GPS.
GBS	Deteção de falhas de satélites GNSS.
GGA	Dados fixos de GPS.
GMP	Dados fixos de projeção de mapas GNSS.
GNS	Dados fixos de GNSS.
GRS	Faixa de resíduos GNSS.
GSA	Diluição de precisão (DOP) e satélites ativos GNSS.
GST	Erro estatístico da pseudo distância ente o satélite e o receptor (GNSS).
GSV	Satélites visíveis GNSS.

**Quadro 2** - Sentenças NMEA relacionadas ao GNSS.  
**Fonte:** Tradução do Quadro 8.5 de EL-RABBANY(2006, p.119).

A estrutura de uma sentença GGA (Dados fixos de GPS) pode ser vista na figura 2.

```
$GPGGA,hhmmss.ss,||||.||,a,yyyyy.yy,a,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx*hh<CR><LF>
$GPGGA,115417.00,4338.123456,N,07938.123456,W,1,10,01.1,095,M,,M,999,000
```

**Figura 2** - Estrutura de uma sentença GGA  
**Fonte:** EL-RABBANY(2006, p.119)

Os elementos da sentença GGA são descritos no Quadro 3.

Símbolo	Descrição
\$	Delimitador do início da sentença.
GP	Identificador do dispositivo transmissor (GPS neste caso).
GGA	Identificador de dados (dados fixos de GPS neste caso).
,	Delimitador do campo de dados.
hhmmss.ss	Tempo da posição no sistema UTC (horasminutossegundos.decimal). Os dígitos decimais são variáveis.
.	Latitude (grausminutos.decimal). Os dígitos decimais são variáveis.
A	N/S (Norte ou Sul)
yyyyy.yy	Longitude (grausminutos.decimal). Os dígitos decimais são variáveis.
A	L/O (Leste ou Oeste).
X	Indicador de qualidade GPS. 0 = Fixação indisponível ou inválida. 1 = Posicionamento pontual com código C/A. 2 = DGPS com código C/A. 3 = Posicionamento Pontual com código P. 4 = Parâmetros ambíguos RTK fixados a valores inteiros. 5 = Parâmetros com ambiguidade flutuante RTK. 6 = Modo de estimativa. 7 = Modo de entrada manual. 8 = Modo de Simulação.
Xx	Número de satélites utilizados na produção da solução.
x.x	Diluição da Precisão Horizontal (HDOP).
x.x	Altitude ortométrica.
M	Metros (unidade da altitude ortométrica).
x.x	Altura geoidal acima do elipsóide WGS84.
M	Metros (unidade da altura geoidal).
x.x	Idade dos dados diferenciais do GPS em segundos.
Xxxx	Identificação da estação de referência diferencial (0000 a 1023).
*	Caractere delimitador da soma de verificação ( <i>checksum</i> ).
Hh	Campo da soma de verificação.
<CR><LF>	Fim da Sentença.

**Quadro 3** - Termos da Sentença GGA  
**Fonte:** Tradução do Quadro 8.6 de EL-RABBANY(2006, p.120)



## 2.2 SISTEMA GLOBAL PARA COMUNICAÇÃO MÓVEL (GSM)

Segundo LIN e CHLAMTAC (2001, p.7), GSM é um sistema celular digital desenvolvido pela Conferência Européia das Administrações de Correios e Telecomunicações (CEPT) e seu sucessor, o Instituto Europeu de Normas de Telecomunicações (ETSI) para oferecer compatibilidade entre serviços de telefonia celular entre diferentes países europeus.

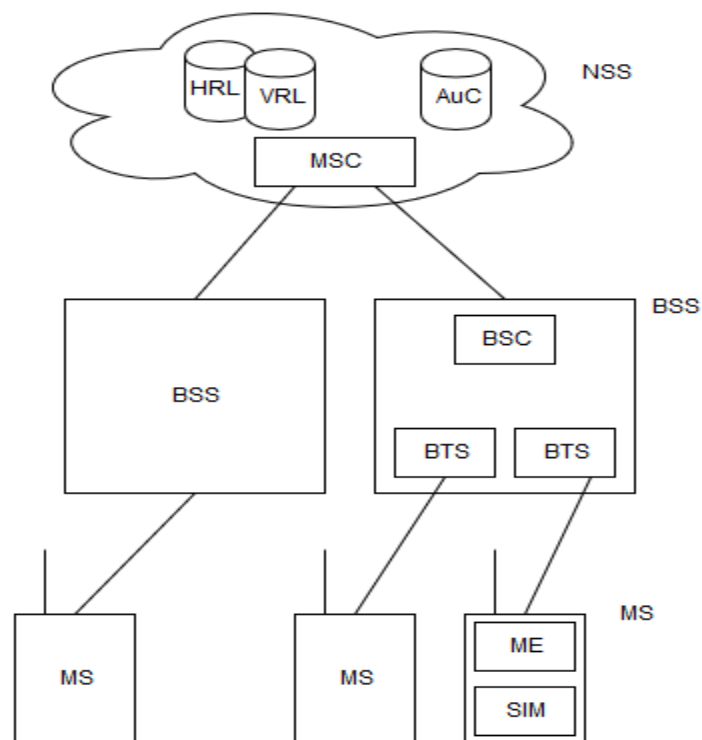
GSM combina acesso múltiplo por divisão de tempo (TDMA) e acesso múltiplo por divisão de frequência (FDMA). Segundo TABBANE(2000, p.83), através do TDMA, que possui taxa de transmissão de informação maior que o FDMA, o hardware do rádio na estação base pode ser compartilhado por múltiplos usuários, uma vez que a portadora é dividida em N intervalos de tempo, e portanto pode ser compartilhada por N terminais.

Em uma estação GSM, cada par de rádios transmissores/receptores suportam oito canais de voz, cada um com uma taxa de codificação de fala de 13 kbps, diferentemente da base do Serviço de Telefonia Móvel Avançado (AMPS) - primeiro sistema de telefonia celular, desenvolvido em 1970 - responsável pela primeira geração de sistemas celulares analógicos, que precisa de um par transmissor receptor para cada canal de voz.

Conforme LIN e CHLAMTAC (2001, p.151), o GSM possui como requisitos básicos prover portabilidade de serviço, ou seja, os telefones móveis podem ser utilizados em todos os países participantes, oferecer chamadas de voz com no mínimo a mesma qualidade dos sistemas analógicos, permitir o uso eficiente do espectro de frequências e coexistir com sistemas anteriores ao GSM, não exigir que redes públicas fixas sejam amplamente modificadas e que os parâmetros do sistema sejam escolhidos de forma a minimizar os custos do sistema completo.

A figura 3 ilustra a arquitetura GSM. Nesta arquitetura, uma Estação Móvel (MS), composta pelo Módulo de Identificação do Assinante (SIM) e o Equipamento Móvel (ME), se comunica com uma Estação Base (BSS), que possui uma Estação de Transmissão/recepção (BTS), com equipamentos específicos para comunicação via interface de rádio com o MS, e uma Estação Base de Controle (BSC),

responsável pela comutação, conectada ao Centro Móvel de Comutação (MSC), geralmente composto de uma rede de comutação fixa padrão a qual algumas funções específicas são adicionadas. O MSC faz parte da Rede e Subsistema de Comutação (NSS), assim como os banco de dados HLR (registro local ao qual a identidade de um MS é associada), VRL (armazena temporariamente informações de assinantes visitantes, isto é, que não possuem registro local) e um centro de autenticação (AuC) .



**Figura 3 - Arquitetura GSM**

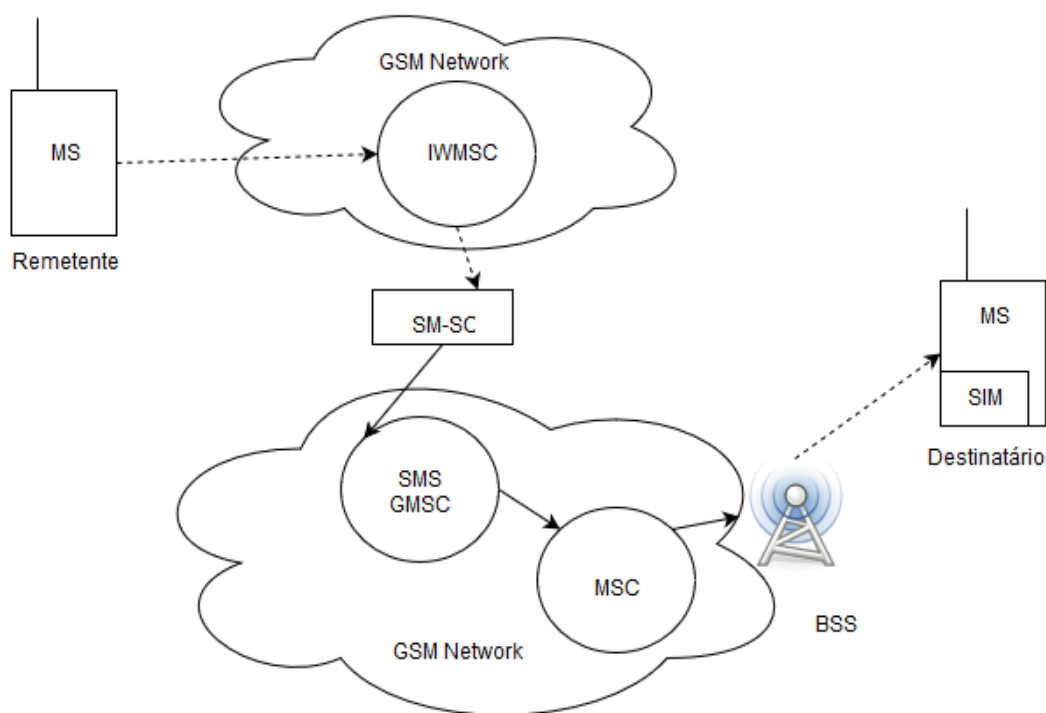
**Fonte:** Adaptado da figura 9.1 de LIN e CHLAMTAC (2001, p.153).

## 2.3 SHORT MESSAGE SERVICE (SMS)

Conforme LIN e CHLAMTAC (2001, p.219), SMS fornece troca de mensagens sem a necessidade de conexão. Cada mensagem pode conter até 160 caracteres do alfabeto padrão GSM, permitindo enviar tamanhos maiores que este através da concatenação de mensagens. Dois tipos de SMS foram definidos: Serviço de

transmissão por célula (*broadcast*), que periodicamente entrega mensagens a todos os inscritos em uma dada área, e serviços ponto a ponto, que enviam mensagens a usuários específicos.

A figura 4 mostra a arquitetura do SMS ponto a ponto. A mensagem é primeiramente enviada pelo remetente da mensagem (MS) ao Centro Móvel de Comutação da rede GSM a qual o MS pertence (IWMSC), que então envia para central SMS (SM-SC). A SM-SC é conectada à rede GSM através de um *gateway* do MSC SMS (SMS GMSC). A SM-SC pode se conectar a diversas redes GSM e diversas SMS GMSCs em uma rede GSM. O SMS GMSC localiza o Centro Móvel de Comutação atual do destinatário da mensagem e encaminha ao MSC encontrado. O MSC transmite a mensagem às estações base (BSS), que entrega a mensagem ao destinatário através de sua Base de Transmissão/Recepção (BTS).



**Figura 4 - Arquitetura SMS**

**Fonte:** Adaptado da figura 12.1 de LIN e CHLAMTAC (2001, p.221).

## 2.4 SOLUÇÕES EXISTENTES

Durante a revisão bibliográfica, foram encontrados três documentos que se relacionam com este projeto. Nesta seção, os trabalhos serão resumidos, e serão analisadas as características em comum com a solução adotada neste documento.

BENTES [13] desenvolveu um sistema de segurança veicular baseado em Arduino, composto de um módulo embarcado no veículo e uma aplicação Web, que permite ao usuário rastrear e bloquear seu veículo.

O controle de acesso da aplicação é feito através de uma tela de *login* e senha, e a interface permite rastrear o veículo pertencente ao usuário de forma contínua, através de um banco de dados relacional trabalhando em conjunto com o API *Google Maps*. O bloqueio do veículo é feito através de uma requisição HTTP ao hardware embarcado, sendo acionado por um botão na interface.

Foram utilizados um *Shield* GPS e outro GSM em conjunto com o Arduino, além de um relé normalmente fechado, colocado em série com a chave de ignição, de forma que, quando acionado, a bomba de combustível deixa de funcionar por falta de alimentação. A elaboração do sistema utilizou conceitos de Engenharia de Software para concepção de plataformas em tempo real, através de diagramas UML para descrever requisitos, arquiteturas e componentes de hardware/software.

O sistema compartilha características com este projeto, mas depende que o usuário esteja perto de um computador com acesso a internet para realizar o bloqueio do veículo, o que não possibilita uma resposta rápida em situações em que o motorista tivesse seu veículo roubado distante de sua residência. O próprio autor reconhece que a utilização do telefone celular traria benefícios ao seu sistema, considerando a hipótese em trabalhos futuros.

BESZCZYNSKI [14] produz uma aplicação semelhante a BENTES [13], utilizando um módulo TELIT que possui GPS integrado, e capacidade de comunicação utilizando GSM/GPRS além de um interpretador de *scripts* da linguagem *Python*.

Foi desenvolvida uma página PHP que permite acompanhar a localização contínua do veículo, através de um banco de dados SQLServer, obter a foto do

motorista através de uma câmera conectada à porta serial do hardware embarcado, e enviá-la através de protocolo FTP além de consultar o status do veículo e exigir seu bloqueio ou desbloqueio. O autor também reconhece que a opção de bloqueio e consulta através de mensagem SMS para o celular seria uma importante extensão do projeto realizado.

GUARAGNI [15] desenvolveu um sistema que monitora a posição de um caminhão tanque, assim como a temperatura e volume da resina por ele transportada. Os dados são enviados através da rede GSM.

Foi utilizada uma plataforma de prototipagem eletrônica Arduino MEGA, um *shield* para cartões de memória do tipo SD de capacidade máxima 4GB (Para armazenar as variáveis), um módulo RTC (REAL TIME CLOCK) para registrar a data e hora em que as variáveis foram registradas, um módulo GPS *Head Studio* para obtenção da posição do caminhão, o módulo SIM900 para comunicação GSM/GPRS, um sensor de temperatura DS18B20 e o módulo HRC-SR04, responsável pela medição de nível da resina através de ultrassom.

A posição do caminhão, a temperatura e o nível da resina no caminhão são armazenados no cartão de memória a cada minuto. Sempre que há uma descarga (abertura da válvula do fundo do tanque), ou quando a temperatura atinge um valor crítico, um SMS é enviado automaticamente a um número pré-definido com a posição do caminhão.

Apesar de não apresentar uma solução para os problemas expostos neste documento, as tecnologias de comunicação e obtenção de posição que foram utilizadas por GUARAGNI [15] são similares às deste projeto. O módulo SIM900 é fabricado pela Shanghai SIMCom wireless solution Ltd. que também produz o SIM908. A comunicação utilizando a rede GSM/GPRS do SIM908 é equivalente àquela feita pelo SIM900.

### 3. DESCRIÇÃO DO PROBLEMA

Neste capítulo o problema é detalhado, e são definidas as situações em que o sistema pode ser utilizado. O modelo do problema foi descrito através de seus requisitos funcionais e não funcionais, para posterior utilização na modelagem da solução.

Segundo o código penal brasileiro [16], furto é definido pelo artigo 155 como: “Subtrair, para si ou para outrem, coisa alheia móvel”, enquanto roubo é definido pelo artigo 157 como: “Subtrair coisa móvel alheia, para si ou para outrem, mediante grave ameaça ou violência a pessoa, ou depois de havê-la, por qualquer meio, reduzido à impossibilidade de resistência”.

Desta forma, o problema possui duas vertentes que devem ser tratadas pela solução adotada. Em uma delas o criminoso adentra o veículo que se encontra parado distante de seu proprietário, em outra, o proprietário é abordado diretamente.

No caso de furto, a solução adotada precisa avisar o usuário que o veículo não se encontra mais na posição em que foi estacionado, e permitir que sua posição atual seja monitorada pelo motorista para que haja uma reposta rápida dele e das autoridades de segurança.

No caso de roubo há risco de vida para o proprietário, e a solução deve tentar minimizar a reação do motorista contra o criminoso, através da garantia de rápida recuperação do veículo.

#### 3.1 MODELO DO PROBLEMA

A modelagem do problema foi feita através do levantamento dos requisitos funcionais e não funcionais que precisam ser atendidos pela solução adotada, descrita no capítulo 4.

### **3.1.1 Requisitos Funcionais Básicos**

O sistema deve permitir que o administrador defina os privilégios do próximo usuário cadastrado, em relação a receber mensagens de que o veículo não está mais parado, obter a posição do veículo e ordem de parada.

O sistema deve verificar se um usuário possui autorização para realizar uma ação.

A posição do veículo deve ser obtida e exibida para o administrador ou usuário autorizado, quando solicitado por este através de mensagem de texto de seu telefone celular.

O veículo deve ser parado quando solicitado por um administrador ou usuário autorizado, através de mensagem de texto de seu telefone celular.

O sistema deve permitir a troca de modos de operação, Passivo ou Ativo, através de mensagem de texto de um dos administradores.

O sistema deve permitir que os administradores indiquem um usuário como motorista atual, para receber notificações sobre a ignição do veículo no modo Ativo.

### **3.1.2 Requisitos Funcionais de Cadastro**

O sistema deve cadastrar o número de telefone associado à primeira mensagem enviada ao cartão SIM vazio como primeiro administrador.

O segundo administrador só deve ser cadastrado pelo primeiro, através de mensagem de texto de seu telefone celular.

Os demais usuários do sistema devem ser cadastrados por um dos dois administradores, através de mensagem de texto.

O sistema não deve permitir tentativas de cadastro de novos usuários por usuários cadastrados sem privilégios de administrador, ou de mais de dois administradores, e deve enviar mensagem de texto avisando ao solicitante que o cadastro não foi efetuado.

### **3.1.3 Requisitos Funcionais de Mensagens**

O sistema deve enviar mensagem de texto ao usuário confirmando que uma ação foi executada.

Uma mensagem de texto deve ser enviada ao usuário caso este não possua autorização para realizar determinada ação.

### **3.1.4 Requisitos Funcionais da Interface Android**

A interface deve ser desenvolvida para versões Android superior a versão 4.0.0.

A interface deve permitir preenchimento automático do conteúdo das mensagens de pedido de Localização e Bloqueio.

O envio de mensagem ao sistema deve ser possível através de um botão exclusivo na interface.

A localização do veículo deve ser mostrada em um mapa na interface quando o pedido for realizado por um administrador ou usuário autorizado através de mensagem de texto de seu telefone celular.

### **3.1.5 Requisitos Não Funcionais**

Não pode haver interferência com outros sistemas dentro do veículo.

O dispositivo instalado no veículo não deve destoar do interior do mesmo.

Na interface, o usuário deve ser capaz de distinguir o conteúdo dos botões de preenchimento automático, através da cor ou formato dos botões.



## 4. MODELO DA SOLUÇÃO

Neste capítulo são definidos o modelo da solução adotada e os componentes de *hardware* e *software* que atendem os requisitos funcionais e não funcionais descritos no capítulo anterior.

A solução adotada dispõe de um hardware no interior do veículo, conectado ao sistema de ignição e à bomba de combustível. No modo Ativo, o sistema considera o carro em movimento quando a ignição é acionada e envia mensagem SMS ao telefone celular dos usuários que possuam privilégio de administrador. O pedido de parada imediata do veículo é realizado pela interrupção do fornecimento de combustível ao motor. Para facilitar a apresentação, os sistemas de ignição e bomba de combustível foram simulados através de computador.

Uma interface gráfica para celulares Android foi desenvolvida para facilitar a comunicação dos usuários com o hardware dentro do veículo, permitindo enviar mensagens SMS pré-definidas, a visualização de mensagens recebidas do hardware embarcado e a posição do veículo em um API de mapas quando sua posição for solicitada pelo usuário.

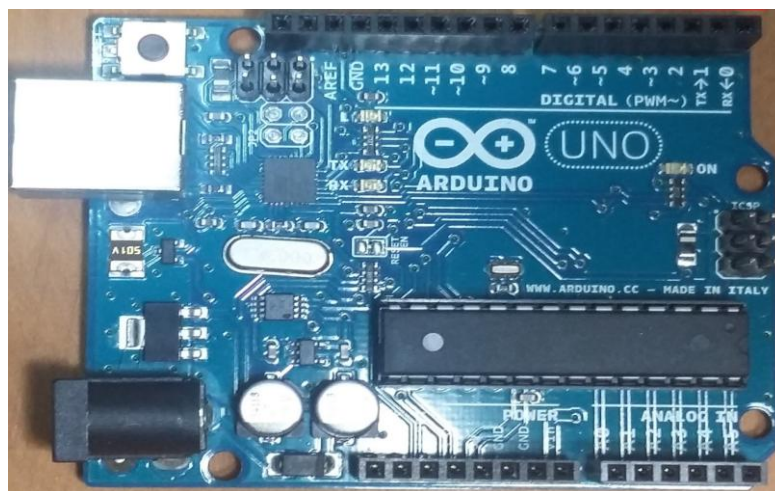
### 4.1 HARDWARE

A solução adotada para satisfazer os requisitos de hardware listados no capítulo anterior dispõe de uma placa de prototipagem eletrônica Arduino UNO, um módulo SIM908 que integra as tecnologias GSM/GPRS e GPS, fabricado pela Shanghai SIMCom wireless solution Ltd., e um *Shield* que possibilita conectar o módulo ao Arduino UNO, às antenas do GPS e da tecnologia GSM, e ao leitor de cartão de telefonia celular. O cartão SIM de telefonia celular, utilizado para armazenar os números de telefone dos usuários do sistema e as mensagens enviadas pelos usuários ao hardware no veículo, é fabricado pela Vivo.

Foram utilizados também uma fonte de tensão contínua de 9 V, com fornecimento máximo de corrente de 1 A, conectada ao DC P4 Fêmea do *Arduino* e um computador conectado à porta serial da placa Arduino, utilizado primeiramente para a programação do sistema e posteriormente para simulação dos sistemas de ignição do veículo e bomba de combustível.

#### 4.1.1 Arduino UNO

Arduino Uno, mostrado na figura 5, é uma plataforma de prototipagem eletrônica com base em microcontroladores ATmega328P, que pode ser conectada ao computador através de uma porta USB, e programada em linguagem padrão C/C++ através de uma IDE própria. Ela faz parte de uma família de placas que diferem na quantidade de portas de entrada/saída analógicas e digitais, assim como na velocidade de seus microcontroladores.



**Figura 5 - Arduino UNO**  
**Fonte:** O Autor

As especificações técnicas da plataforma Arduino Uno, obtidas do site do fabricante [7], são mostradas no Quadro 4. O ATmega328P vem pré-programado com um *bootloader* que permite enviar novos códigos à plataforma sem necessidade de um *hardware* programador externo.

Microcontrolador	ATmega328P
Tensão de Operação	5V
Tensão de Entrada (recomendada)	7-12V
Tensão de Entrada (limite)	6-20V
Pinos E/S Digitais	14, dos quais 6 produzem saída MLP
Pinos E/S Digitais de MLP	6
Pinos de Entrada Analógicos	6
Corrente DC por pino de E/S	20 mA
Corrente DC para o Pino 3.3V	50 mA
Memória Flash	32 kB (ATmega328P), dos quais 0,5 kB são usados pelo <i>bootloader</i>
SRAM	2 kB (ATmega328P)
EEPROM	1 kB(ATmega328P)
<i>Clock</i>	16 MHz
Altura	68,6 mm
Largura	53,4 mm
Peso	25 g
* MLP: Modulação por Largura de Pulso	

**Quadro 4** - Especificações Técnicas do Arduino Uno  
**Fonte:** Tradução de <https://www.arduino.cc/en/Main/ArduinoBoardUno>

As plataformas da família Arduino possibilitam o uso de sensores, módulos (versões reduzidas de placas) e *Shields* (elementos que podem ser conectados à plataforma, fornecendo novas características), o que muitas vezes dispensa o uso de soldas e aumenta a praticidade do projeto. O esquema elétrico do Arduino Uno está disponível no Anexo A.

#### 4.1.2 Módulo SIM908

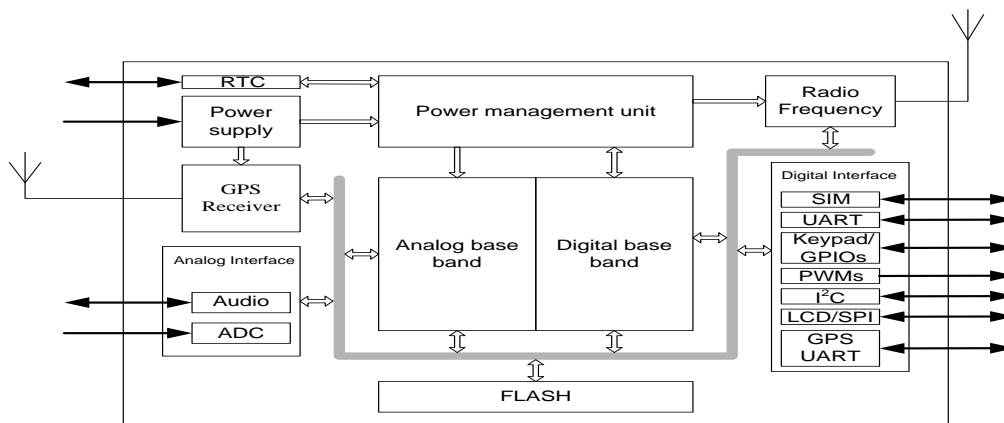
O SIM908, mostrado na figura 6, é um módulo de comunicação GSM/GPRS *quadband* com pilha TCP/IP que possui GPS integrado. Ele é fabricado pela

Shanghai SIMCom wireless solution Ltd. e foi escolhido para este projeto pela praticidade que proporciona ao integrar estas tecnologias. Módulos anteriores ao SIM908 como o SIM900, do mesmo fabricante, não possuem GPS integrado, o que obrigaria o uso de um módulo adicional para obtenção da posição do veículo.



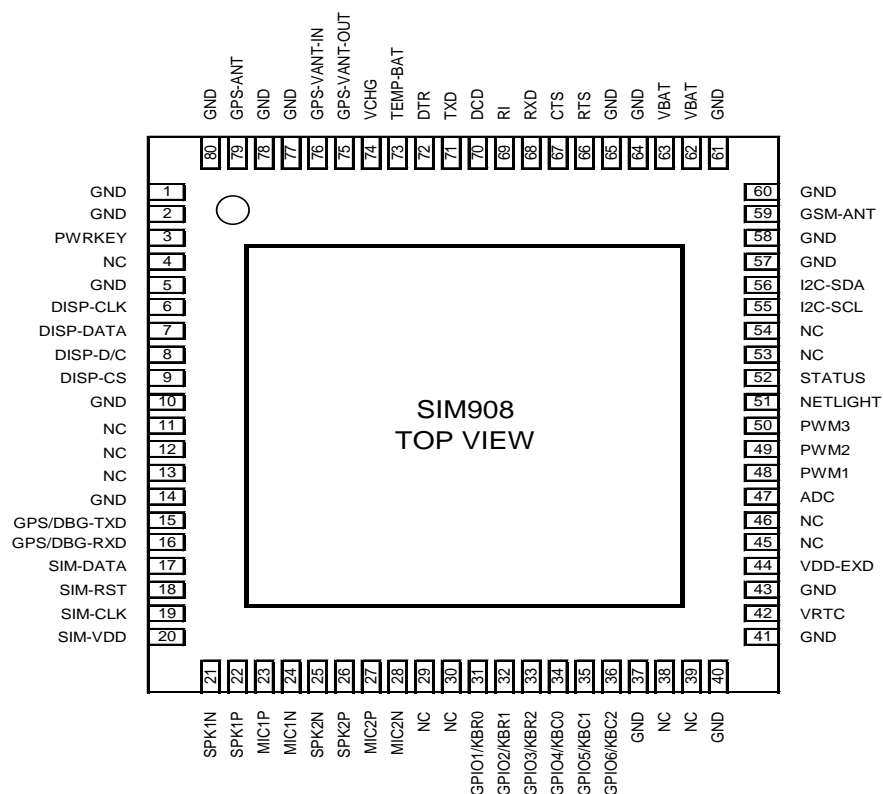
**Figura 6 - Módulo SIM908**  
Fonte: O Autor

O diagrama funcional fornecido pelo fabricante é mostrado na Figura 7. O SIM908 possui uma interface de áudio e um conversor analógico digital (ADC) auxiliar que pode ser utilizado para medir tensão, porém não foram utilizados neste projeto.



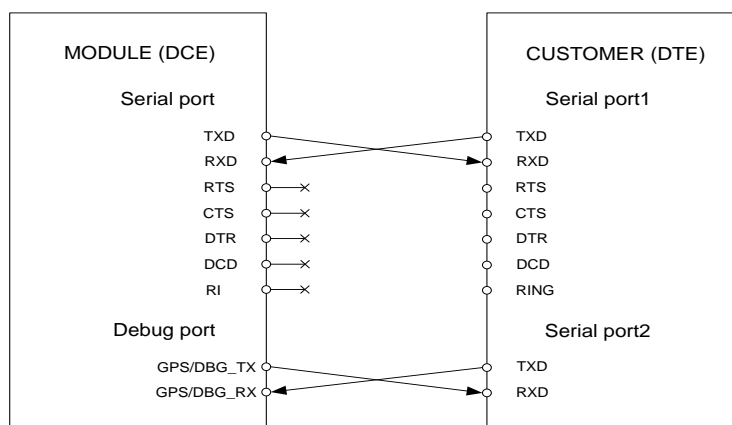
**Figura 7 - Diagrama Funcional do SIM908**  
Fonte: SIM908 Hardware Design, p. 12.

A pinagem do SIM908 é mostrada na Figura 8. A comunicação do módulo com o Arduino Uno é feita pelos pinos 71 (TXD) e 68 (RXD), do SIM908, de transmissão serial. Os pinos 15 (GPS/DBG-TXD) e 16 (GPS/DBG-RXD) são utilizados para obtenção do código NMEA do GPS, podendo ser utilizados para atualização de software e depuração.



**Figura 8** - Pinagem do módulo SIM908  
**Fonte:** SIM908 Hardware Design, p. 13

As conexões do módulo Sim908 (DCE) com o Arduino Uno (DTE) são indicadas na Figura 9. Os pinos 0 e 1 do Aduino foram utilizados, respectivamente, como RXD e TXD da porta Serial 1. Os pinos 2 e 3 do Arduino foram utilizados, respectivamente, como TXD e RXD da porta Serial 2 (Obtenção do código NMEA).



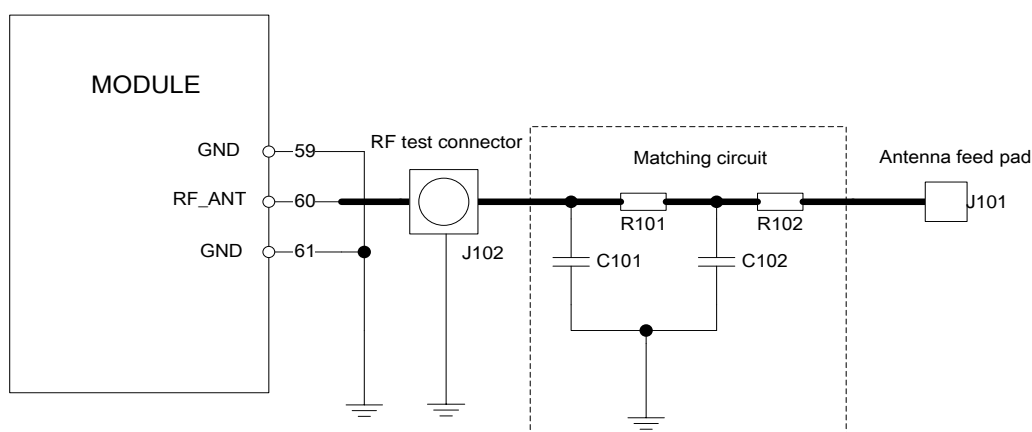
**Figura 9** - Conexão entre Arduino e SIM908  
**Fonte:** SIM908 Hardware Design, p. 29.

Para este projeto foi utilizada a tecnologia GSM para comunicação do hardware embarcado com o telefone celular. O SIM908 possui quatro frequências de operação para esta tecnologia, que são pesquisadas automaticamente: GSM, Enhanced GSM (EGSM), Digital Cellular Service (DCS) e Personal Communication System (PCS), que possuem as faixas de frequência mostradas no Quadro 5.

Frequência	Recebe	Transmite
GSM850	869 ~ 894MHz	824 ~ 849 MHz
EGSM900	925 ~ 960MHz	880 ~ 915MHz
DCS1800	1805 ~ 1880MHz	1710 ~ 1785MHz
PCS1900	1930 ~ 1990MHz	1850 ~ 1910MHz

**Quadro 5** - Frequências de Operação do SIM908.  
**Fonte:** Tradução de SIM908 Hardware Design, p. 48.

O SIM908 envia e recebe mensagens SMS do tipo *Mobile Originated* (MO), *Mobile Terminated* (MT), *Cell Broadcast* (CB), *Texto* e *modo PDU* (Informação no formato binário), armazenando-as em um cartão SIM de 1,8V ou 3,0V. A interface da antena utilizada no envio de mensagens SMS é mostrada na Figura 10.



**Figura 10** - Interface da antena GSM com o SIM908  
**Fonte:** SIM908 Hardware Design, p. 39.

Os modos de operação do SIM908 são detalhados no Quadro 6.

Modo	Função	
Desligado	Desligamento pelo comando “AT+CPOWD=1” ou usando PWRKEY. Apenas a fonte de alimentação para o RTC é mantida. Software não estará ativo. A porta serial não estará acessível. Fonte de alimentação (conectada ao VBAT) continua aplicada.	
Modo de Funcionalidade Mínima	Comando “AT+CFUN” pode ser utilizado para ajustar o módulo para funcionalidade mínima sem remover a fonte de alimentação. Neste modo, a parte RF do módulo não irá funcionar ou o cartão SIM não estará acessível, ou os dois estarão fechados e a porta serial ainda estará acessível.	
Modo de Carregamento	O módulo vai automaticamente para este modo quando um carregador e bateria são conectados a um SIM908 desligado. Neste modo, o módulo não procura por redes e possui acesso limitado a comandos AT.	
Modo de Carregamento durante Operação Normal	O módulo vai automaticamente para este modo quando um carregador é conectado em modo de operação Normal, quando a tensão da bateria não é menor que 3.2V.	
Operação Normal	GSM/GPRS SLEEP	Módulo entra automaticamente neste modo se não houver interrupções de hardware ou dados recebidos da rede. Neste modo, o módulo ainda pode receber mensagens SMS.
	GSM IDLE	Software está ativo. Módulo registrado à rede GSM e pronto para comunicação.
	GSM TALK	Conexão entre dois assinantes em progresso. Neste caso, o consumo de energia depende da configuração da rede.
	GPRS STANDBY	Módulo está preparado para dados GPRS, mas nenhum dado está sendo enviado ou recebido atualmente. Neste caso, o consumo de energia depende das configurações de rede e da configuração do GPRS.
	GPRS DATA	Há transferência de dados GPRS (PPP ou TCP ou UDP) em progresso. Neste caso, o consumo de energia está relacionado com a configuração da rede.

**Quadro 6 - Modos de Operação do SIM908**  
**Fonte:** Tradução de SIM908 Hardware Design, p. 11.

A lista de comandos AT aceita pelo módulo GPS e suas descrições são mostradas no Quadro 7.

Comando	Descrição
AT+CMGD	Exclui mensagem SMS.
AT+CMGF	Seleciona o formato da mensagem SMS.
AT+CMGL	Lista as mensagens SMS.
AT+CMGR	Leitura da mensagem SMS recebida.
AT+CMGS	Envia mensagem SMS.
AT+CMGW	Escreve mensagem SMS para a memória.
AT+CMSS	Envia mensagem SMS armazenada.
AT+CNMI	Indicadores de nova mensagem SMS.
AT+CPMS	Definição do Armazenamento da mensagem SMS.
AT+CRES	Restaurar configurações SMS.
AT+CSAS	Salva configurações SMS.
AT+CSCA	Endereço da Central de serviços SMS.
AT+CSCB	Seleciona transmissão SMS por célula ( <i>Broadcast</i> ).
AT+CSDH	Mostra os parâmetros do SMS no modo texto.
AT+CSMP	Configura os parâmetros do SMS no modo texto.
AT+CSMS	Seleciona o serviço de mensagens.

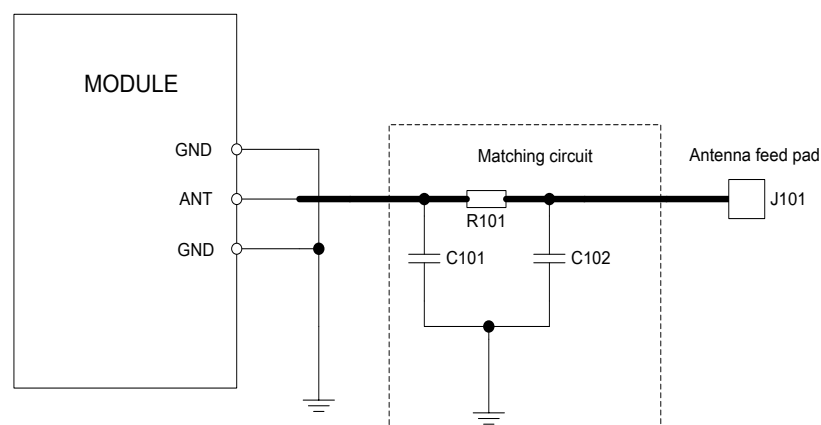
**Quadro 7** - Comandos AT do GSM

**Fonte:** Tradução de SIM908 Hardware Design, p. 40.

O GPS integrado ao SIM908, utilizado para obter a posição do veículo, possui exatidão da posição horizontal de dois metros e meio. Ele possui receptor com quarenta e dois canais e com taxa de atualização de 1 Hz. Quando o GPS é executado, a interface GSM deve estar ligada, uma vez que ela é responsável por seu controle.

Os protocolos utilizados são o NMEA e *One Socket Protocol* (OSP). O código NMEA é obtido pela porta *Debug*: GPS/DBG\_TXD (pino 15) e GPS/DBG\_RXD (pino 16), com taxa de transmissão padrão de 115200 bps. A interface da antena do GPS, responsável pela sincronização com os satélites e obtenção da posição é mostrada na Figura 11.





**Figura 11** - Interface da antenna GPS com o SIM908  
**Fonte:** SIM908 Hardware Design, p. 41.

Todas as funções do GPS são controladas por comandos AT, via porta serial, assíncrona, de taxas de transmissão: 1200, 2400, 4800, 9600, 19200, 38400, 57600 e 115200 bps. Ela dispõe ainda da função *autobauding*, que suporta taxas de transmissão entre 1200 a 57600 bps.

Na comunicação entre o Arduino e o SIM908, o caractere 'A' é enviado primeiramente, do Arduino para o módulo SIM908, para sincronizar a taxa de transmissão. É recomendável enviar "AT" continuamente, até que a resposta "OK" seja recebida pela porta serial do Arduino, que deve ser configurada para receber dados de 8 bits, sem paridade e com um bit de parada.

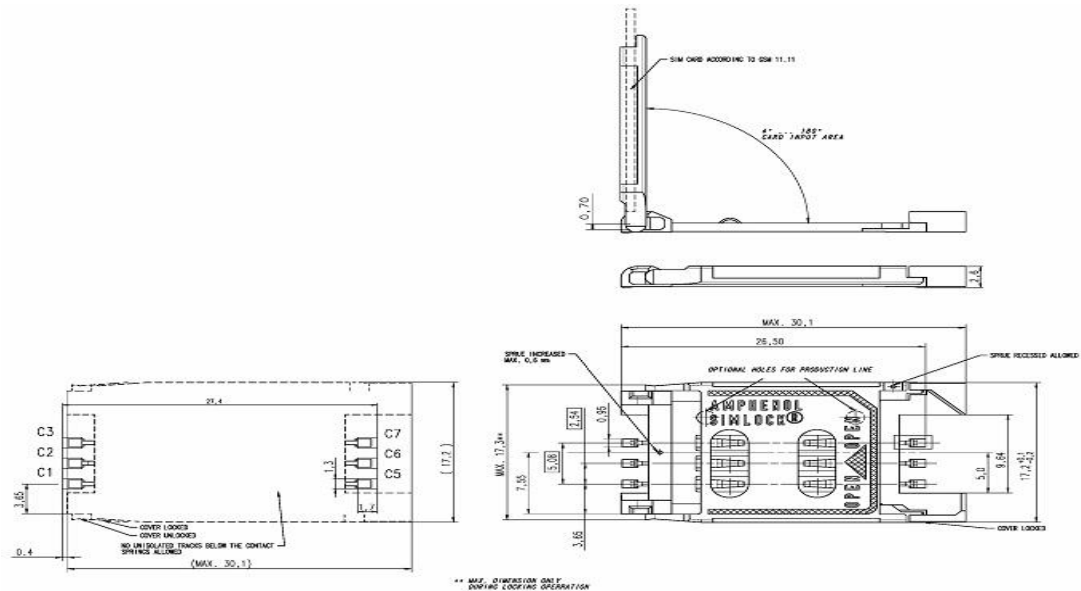
A lista de comandos AT aceita pelo GPS e suas descrições são mostradas no Quadro 8.

Comando	Descrição
AT+CGPSPWR	Liga/Desliga o GPS.
AT+CGPSRST	Modo do Reset do GPS.
AT+CGPSSTATUS	Obtém Estado atual do GPS.
AT+CGPSOUT	Controle da saída de dados NMEA do GPS.
AT+CGPSINF	Obtém informação sobre a posição atual do GPS.
AT+CGPSIPR	Ajusta a taxa de transmissão da saída serial do código NMEA do GPS.

**Quadro 8** - Comandos AT do GPS  
**Fonte:** Tradução de SIM908 Hardware Design, p. 40.

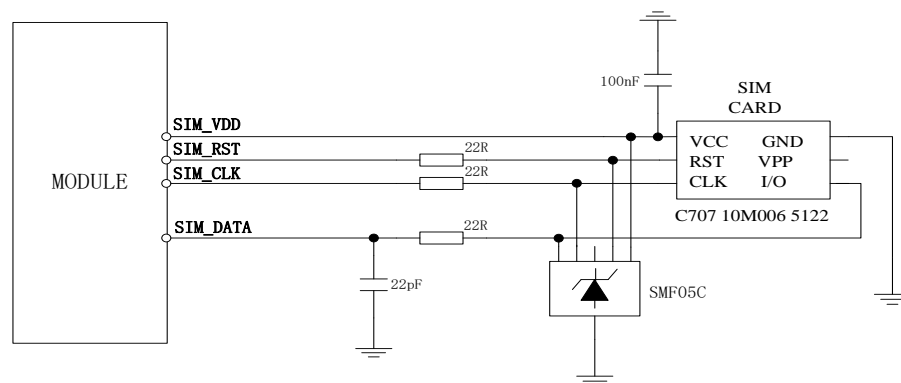
No Anexo B estão disponíveis características elétricas adicionais do SIM908 para o entendimento e manutenção do módulo.

### 4.1.3 Leitor do Cartão SIM



**Figura 12 - Leitor Cartão SIM de 6 Pinos**  
**Fonte:** SIM908 Hardware Design, p. 34.

O leitor de cartão SIM do *Shield* tem as dimensões especificadas na figura 12 e possui seis pinos que devem ser conectados ao módulo SIM908 de acordo com a figura 13. Ele é compatível com as especificações do GSM fase 1, GSM fase 2 e cartão SIM de 64 kbps, de 1,8V ou 3,0V.



**Figura 13 - Conexão do Leitor de Cartão SIM ao módulo SIM908**  
**Fonte:** SIM908 Hardware Design, p.34.

No Quadro abaixo, os nomes dos pinos e suas respectivas descrições são especificadas pelo fabricante.

Nome do Pino	Sinal	Descrição
C1	SIM-VDD	Alimentação do cartão SIM.
C2	SIM-RST	Reset do cartão SIM.
C3	SIM-CLK	<i>Clock</i> do cartão SIM.
C4	GND	Conectado ao Terra.
C5	GND	Conectado ao Terra.
C6	VPP	Não conectado.
C7	SIM-DATA	Dados de E/S do Cartão SIM.
C8	SIM-PRESENCE	Detecta a presença do cartão SIM.

**Quadro 9** - Pinos do módulo sim908 conectados ao Leitor de Cartão SIM  
**Fonte:** Tradução de SIM908 Hardware Design, p. 34.

Para o cadastro dos usuários do sistema e seus respectivos privilégios, foi necessário criar uma lista de telefones no cartão SIM. O módulo SIM908 permite criar a lista nos padrões *SIM card* (SM), *Fixed Dial Numbers* (FD), *Last Dialed* (LD), *Received Numbers* (RC), *Own Numbers* (ON) e *Missed Numbers* (MC).

#### 4.1.4 Integração do Hardware

O *shield* utilizado neste projeto facilita a integração do módulo com o Arduino Uno, com o leitor de cartão SIM e possui entrada para as antenas de GPS e SMS.

Neste projeto, o *shield* foi conectado diretamente ao Arduino, como mostrado na Figura 14, porém para realizar a conexão elétrica do Arduino Uno com a bomba de combustível e o sistema de ignição em uma aplicação não simulada, o *shield* teria que ser conectado através de uma *protoboard*, uma vez que, apesar de se conectar a todos os pinos do Arduino, ele somente utiliza os pinos de alimentação, os pinos 0, 1, 2 e 3 de transmissão serial e os pinos 5 e 6 para controle do LED integrado.



**Figura 14** - Conexão do *Shield* com o Arduino Uno

**Fonte:** O Autor

A Figura 15 mostra a conexão das antenas de GPS e SMS e da alimentação de 9V.



**Figura 15** - Integração do Hardware

**Fonte:** O Autor

## 4.2 Software

A interface para o celular foi desenvolvida para dispositivos com versões Android superiores a 4.0.0, através do ambiente Android Studio versão 1.3.2, utilizando *Java Development Kit* (JDK) versão 8 update 5, necessário para programação em linguagem Java. Também foram utilizados o *Software Development Kit* versão 24.3.4 do Android, que fornece as ferramentas necessárias para desenvolver, testar e depurar aplicativos Android e o *Plugin Android Development Kit* (ADT), que permite usar as ferramentas do SDK do Android para desenvolver aplicativos Android.

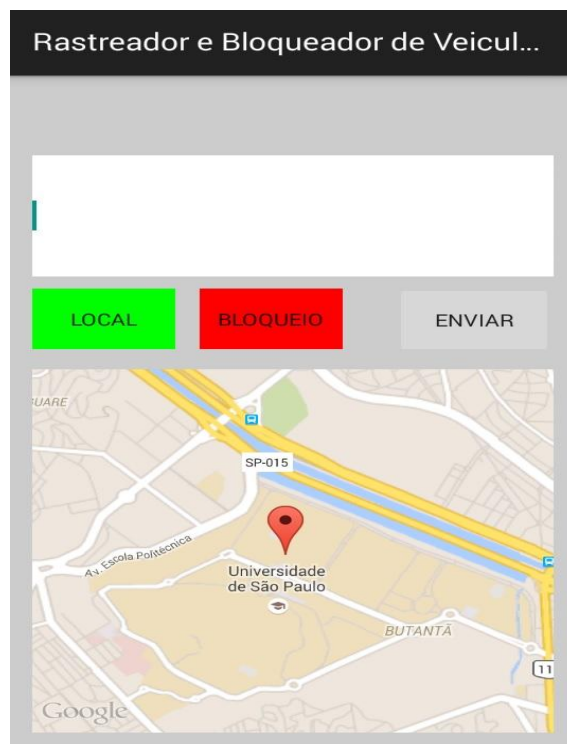
Para o desenvolvimento do programa executado na placa Arduino e comunicação serial da placa Arduino com o computador foi utilizado o IDE Arduino versão 1.6.5 que utiliza linguagem própria, baseada em C e C++, e permite a obtenção e envio de dados através de um cabo USB.

### 4.2.1 Interface Gráfica

A interface gráfica, mostrada na figura 16, possui duas regiões distintas, sendo uma delas para manipulação, envio e recebimento de mensagens SMS e outra destinada à visualização pelo usuário da posição de seu veículo em uma API de mapas.

Ao usuário é fornecida a opção de botões de preenchimento automático para obter a localização do veículo e exigir seu bloqueio, e são distinguíveis pela cor e rótulo. A mensagem só é enviada quando o usuário pressiona o botão Enviar.

Para exibir a posição do veículo, a interface possui o API Google Maps, que mostra automaticamente a posição obtida pelo hardware dentro do veículo e enviada ao celular através de mensagem SMS.



**Figura 16 - Interface Android**  
**Fonte: O Autor**

A interface possui duas classes: SMSSender, representada na figura 17, responsável pelo envio da mensagem de texto e interação com o API Google Maps e SMSReceiver, responsável por receber a mensagem de texto e analisar seu conteúdo.

SMSSender
+tagEditText: EditText +trackButton: Button +blockButton: Button +sendButton: Button +latitude: double +longitude: double +mensagem: String +mensagens: ArrayList<String> +smsMgr: SmsManager
+onCreate(savedInstanceState: Bundle) +setTexto(Texto:String) +onResume() +setUpMapIfNeeded() +setUpMap() +enviarSMS(smsTexto:String)

**Figura 17 - Classe SMSSender**  
**Fonte: O Autor**

Os atributos latitude e longitude armazenam os valores recebidos pela classe SMSReceiver, e são utilizados no método `setUpMap()` e `setUpMapIfNeeded()`, de uso do API Google Maps, para fornecer a localização do veículo no mapa da interface. Os atributos mensagem, mensagens e smsMgr são utilizados no método `enviarSMS(smsTexto: String)` para permitir o envio de mensagens SMS superiores a 160 caracteres através da concatenação de mensagens em um *ArrayList*.

O Apêndice B reproduz o arquivo *Manifest.xml*, que descreve os elementos gráficos que compõem a interface e as permissões que devem ser concedidas pelo usuário e o Apêndice C reproduz a classe SMSReceiver.

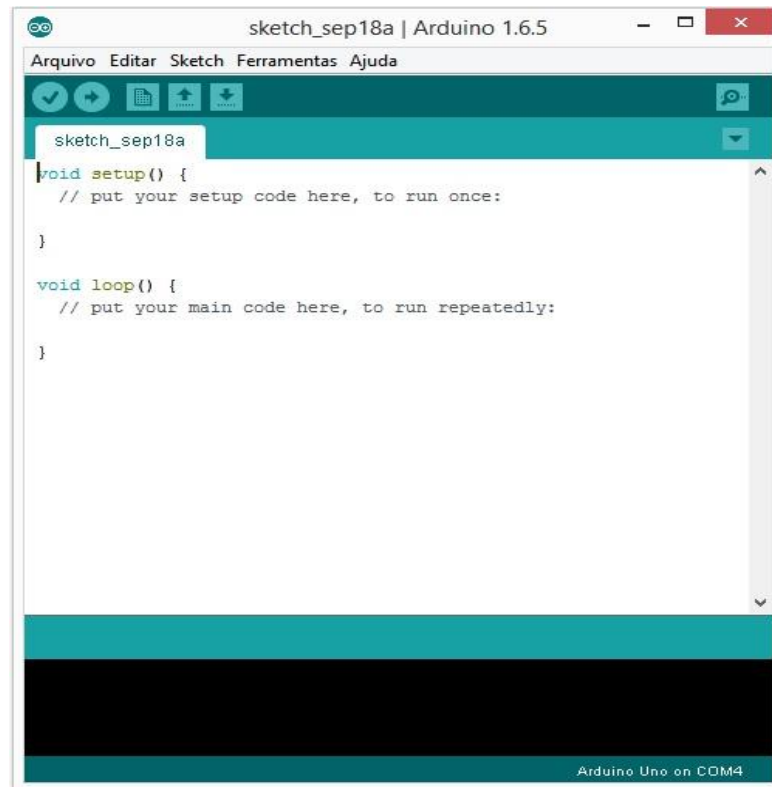
#### 4.2.2 IDE Arduino

O *Arduino Software (IDE)*, mostrada na Figura 18, consiste em um editor de textos, onde são escritos os códigos (chamados de *sketches*) a serem executados pela plataforma, uma área dedicada a mensagens destinadas ao usuário na parte inferior, uma barra de ferramentas que permite compilar o código e transferi-lo para a plataforma Arduino (através do *bootloader* do ATmega328P), assim como botões para criar, abrir ou salvar *sketches*.

O IDE também possui uma barra de menus, que permite manipular arquivos, editar, verificar e compilar o *sketch* atual e também selecionar a porta serial a qual a plataforma Arduino está conectada, e interagir com ela através do Monitor Serial por comandos AT.

Os *sketches* são escritos no editor de texto e salvos com a extensão “.ino”. Eles possuem duas funções principais: *void setup()*, que é executada somente uma vez e configura os elementos da plataforma que serão utilizados e *void loop()*, que é executada continuamente, e corresponde às ações que os elementos da plataforma configurados anteriormente devem realizar.

Bibliotecas podem ser importadas, proporcionando funcionalidades adicionais. Elas devem ser incluídas na estrutura do código, antes da função *void setup()*, através do comando *#include*. Também antes desta função devem ser declaradas as variáveis utilizadas no *sketch*.



**Figura 18** - Software Arduino (IDE)

**Fonte:** O Autor

### 4.3 Testes

Os testes foram realizados com o objetivo entender o funcionamento do SIM908 e sua resposta em relação aos comandos AT para obtenção do código NMEA, envio e recebimento de mensagens de texto, armazenamento e manipulação de contatos telefônicos no cartão SIM e averiguar o correto funcionamento do sistema integrado.

Os códigos utilizados nos testes estão disponíveis no CD que acompanha a monografia, com estrutura descrita no Apêndice E, e as respostas do SIM908 aos testes realizados estão disponíveis no Apêndice A.

### Comentários Bibliográficos

A interface foi programada seguindo o roteiro sugerido por DEITEL [6], adaptado ao IDE Android Studio e às necessidades do projeto. As informações obtidas em sites especializados também foram de fundamental importância na programação do Arduino e do sistema Android.



## 5. ESTUDO DE CASO

Neste capítulo é detalhada a estrutura das mensagens que são interpretadas pelo sistema, para obter a posição do veículo, efetuar seu bloqueio, efetuar o cadastro de novos usuários e definir seus privilégios. Também são detalhados os modos Ativo e Passivo, e as mensagens de resposta do sistema.

No APÊNDICE D foi disponibilizado o fluxograma completo das mensagens, que ilustra a lógica do recebimento e tratamento das mensagens SMS por parte do hardware embarcado, descrita neste capítulo.

### 5.1 Cadastro do Administrador com o cartão SIM vazio

O primeiro usuário a enviar SMS ao hardware embarcado com o cartão SIM vazio, com a mensagem **CADASTRO** será cadastrado automaticamente como administrador do sistema.

### 5.2 Cadastro de Usuário Adicional e Atribuição de Privilégios

Um novo usuário será cadastrado somente pelo administrador do sistema através da mensagem SMS (**CADASTRAR**, **<TELEFONE>**, **<PRIVILEGIO>**), sem os parênteses, onde **<TELEFONE>** corresponde ao número que se deseja cadastrar, no formato internacional, e **<PRIVILÉGIO>** determina qual privilégio o novo usuário terá dentro do sistema.

Caso o novo usuário deva possuir privilégio de administrador, o campo **<PRIVILÉGIO>** será substituído por **ADM** na mensagem. Privilégios individuais podem ser oferecidos, como obtenção da posição pelo GPS, substituindo o campo **<PRIVILÉGIO>** por **GPS** ou bloqueio do veículo, substituindo **<PRIVILÉGIO>** por

**BLOCK.** Se um usuário recebe privilégio de bloqueio do veículo, ele necessariamente obtém privilégio para obtenção da posição do veículo.

### 5.3 Retirada de Usuário Cadastrado

Um administrador pode solicitar a retirada de um usuário, ou de outro administrador do sistema através da mensagem SMS (**RETIRAR, <TELEFONE>**), sem parênteses, onde **<TELEFONE>** corresponde ao número já cadastrado no formato internacional.

### 5.4 Pedido da Posição do Veículo pelo Usuário

O usuário, com privilégios suficientes, pode exigir a posição do veículo a qualquer momento, através da interface em seu telefone celular. É oferecida a opção de preenchimento automático através de um botão na interface do conteúdo da mensagem: **LOCAL**, que deve ser enviada ao hardware embarcado. Para que a mensagem seja enviada, o botão ENVIAR deve ser pressionado.

### 5.5 Pedido de Bloqueio do Veículo pelo Usuário

O usuário que possui privilégio de bloqueio ou de administrador pode exigir a interrupção do fornecimento de combustível ao motor do veículo através do mesmo procedimento descrito no item 5.4, somente modificando o conteúdo da mensagem por: **BLOQUEIO**. Os botões de preenchimento automático são visualmente distintos para que o usuário tenha facilidade em usar a interface.

## 5.6 Modo Ativo e Modo Passivo

No modo ativo, o hardware embarcado monitora o sistema de ignição do veículo e, quando a ignição é acionada, envia a mensagem de texto: **VEICULO EM MOVIMENTO** ao usuário cadastrado com privilégio de administrador, e ao usuário designado por um dos administradores como motorista atual através da mensagem SMS (**MOTORISTA,<TELEFONE>**), sem parênteses, onde **<TELEFONE>** corresponde ao número já cadastrado no formato internacional.

Este modo é destinado ao caso em que o veículo deve ficar estacionado por um longo período de tempo, distante do motorista, por ser uma condição propensa ao furto de veículos.

O modo passivo é destinado ao caso em que o usuário precisa realizar paradas curtas durante seu trajeto, e não deseja ser avisado todas as vezes que o sistema de ignição for acionado.

O administrador pode solicitar a mudança de modo de operação, enviando mensagem de texto: **ATIVO** ou **PASSIVO** ao hardware no interior do veículo.

## 5.7 Mensagens de Resposta

O sistema envia resposta positiva **OK** quando um comando for executado com sucesso, e resposta **ERRO** caso não consiga realizar uma ação.

No caso específico em que um usuário não cadastrado tenta cadastrar um novo número de telefone, ou quando um administrador tenta cadastrar um terceiro administrador, a mensagem enviada pelo hardware embarcado ao usuário será: **NUMERO NÃO CADASTRADO.**

A resposta: **PRIVILEGIOS INSUFICIENTES** é gerada quando um usuário cadastrado tenta realizar uma ação que necessite de privilégios superiores aos que ele possui.

## **6. CONSIDERAÇÕES FINAIS**

Neste capítulo é discutido se os objetivos foram cumpridos, se os requisitos funcionais e não funcionais foram respeitados, assim como as dificuldades encontradas durante o projeto. Também são discutidas as contribuições geradas pelo projeto, e quais as possibilidades futuras de ampliação e inovação do que foi desenvolvido.

### **6.1 CUMPRIMENTO DOS OBJETIVOS**

O módulo SIM908 permitiu realizar a comunicação via SMS com o celular do usuário, e obter a posição do veículo através de seu GPS integrado. Foi possível realizar o cadastro de usuários e o sistema de prioridades através da programação do Arduino e a utilização do cartão SIM como armazenamento.

O fato da antena do GPS não poder ser utilizada em ambientes fechados, e possuir somente um cabo RG-74 de 3 metros de comprimento se demonstrou um desafio para a execução do projeto. Para a apresentação, foi utilizado um cabo extensor de 15 metros RG-58, de mesma impedância do cabo da antena, mas com menor perda de sinal. Para uma aplicação não simulada, a mesma antena utilizada neste projeto pode ser instalada no painel do veículo ou em seu exterior.

Outra dificuldade encontrada foi a limitação de memória do Arduino Uno. Aplicações que necessitem de características adicionais às deste projeto, como obtenção de dados de sensores, devem considerar o uso de memória externa ou de plataformas com maior quantidade de memória interna. A própria família de dispositivos Arduino possui plataformas com maior memória interna como é o caso do Arduino Mega, Arduino Due e Arduino Zero, cujas especificações podem ser encontradas no site do fabricante [17].

## **6.2 CONTRIBUIÇÕES DO TRABALHO**

Como discutido no capítulo 2, as alternativas similares encontradas durante o desenvolvimento deste projeto tinham como planejamento futuro ampliar o sistema através de comunicação do veículo rastreado com dispositivos móveis. A escolha da tecnologia GSM, em particular o SMS, faz uso de uma rede de telefonia estável e de grande porte, o que possibilita o rastreamento e interação com veículos mesmo a longas distâncias, sem a necessidade de conexão a internet.

Apesar de o sistema ter sido desenvolvido especificamente para o rastreamento e bloqueio de veículos, a troca de mensagens entre o aparelho de telefonia celular e a plataforma de prototipagem eletrônica pode ser utilizada para outras finalidades, uma vez que o Arduino possui uma grande quantidade de sensores e módulos compatíveis. O sistema pode ser adaptado para aplicações de automação e controle, segurança domiciliar ou mapeamento da rota de veículos de carga ou marinhos.

## **6.3 TRABALHOS FUTUROS**

Adicionalmente ao que foi desenvolvido neste projeto, novas funcionalidades podem ser implementadas tanto para melhorar o sistema de rastreamento e bloqueio de veículos quanto para as aplicações discutidas no item anterior que possam se beneficiar deste sistema.

O processo de rastreamento pode ser ampliado, de forma a ser executado continuamente após o usuário enviar mensagem ao hardware embarcado, mapeando a rota do veículo.

A troca de mensagens, em conjunto com sensores e módulos adicionais conectados ao Arduino, pode fornecer ao proprietário informações como a quantidade de combustível restante, ou obter a imagem e som do interior do veículo.

Estas informações, em conjunto com informações sobre o local onde o veículo se encontra, obtidas através do API Google Maps, podem gerar sugestões, do sistema para o usuário, dos melhores pontos para se acionar o bloqueio do veículo, de forma

a não prejudicar o tráfego de veículos no local e possibilitar uma resposta mais eficiente das forças de segurança.

Além disso, bancos de dados oferecem ao proprietário a possibilidade de monitorar quais rotas são mais eficientes e estabelecer relações como, por exemplo, o consumo de combustível associado a um motorista em um determinado período de tempo.

#### **6.4 CONCLUSÕES**

As tecnologias escolhidas possibilitaram atingir os objetivos deste projeto, oferecendo uma solução para o problema descrito no capítulo 3, de fácil utilização pelo usuário e que expande os conceitos de trabalhos similares.

O entendimento do funcionamento das tecnologias envolvidas e das limitações do hardware escolhido possibilita a sua utilização em aplicações semelhantes às deste projeto ou que adicionem novas funcionalidades.

## 7. REFERÊNCIAS

- [1] KAPLA; Elliott, HEGARTY; Christopher. **Understanding GPS: Principles and Applications**. Massachusetts: Artech House, 2006.
- [2] National Coordination Office for Space-Based Positioning, Navigation, and Timing, **New Civil Signals**, Disponível em:  
<http://www.gps.gov/systems/gps/modernization/civilsignals/>  
Acessado em: 20/10/2015
- [3] EL-RABBANY, Ahmed. **Introduction to GPS: The Global Positioning System**. Massachusetts: Artech House, 2006.
- [4] TABBANE, SAMI. **Handbook of Mobile Radio Networks**. Massachusetts: Artech House, 2000.
- [5] LIN, Yi-Bing; CHLAMTAC, Imrich. **Wireless and Mobile Network Architectures**. Nova Iorque: JOHN WILEY & SONS, INC., 2001.
- [6] DEITEL, Abbey; DEITEL, Harvey; DEITEL, Paul; MORGANO, Michael. **Android para programadores: Uma abordagem baseada em aplicativos**. Tradução de João Eduardo Nóbrega Tortello; Revisão técnica de Daniel Antonio Callegari. Porto Alegre: Bookman, 2013.
- [7] Arduino, **Arduno Uno**, Disponível em:  
<https://www.arduino.cc/en/Main/ArduinoBoardUno> Acessado em: /01/09/2015
- [8] Shangai SIMCom wireless solution Ltd., **SIM908 Hardware Design**. Shangai, 2011. v. 1.0.
- [9] Shangai SIMCom wireless solution Ltd., **SIM908 AT Command Manual**. Shangai, 2011. v. 1.01.
- [10] SSP/SP, **Taxa de Delito**, Disponível em:  
<http://www.ssp.sp.gov.br/novaestatistica/Pesquisa.aspx>.  
Acessado em: 01/09/2015.
- [11] SSP/SP, **Produtividade Policial**, Disponível em:  
<http://www.ssp.sp.gov.br/novaestatistica/Pesquisa.aspx>.  
Acessado em 01/09/2015.
- [12] SSP/SP, **Ocorrências Policiais Registradas por Mês**, Disponível em:  
<http://www.ssp.sp.gov.br/novaestatistica/Pesquisa.aspx>. Acessado em: 01/09/2015.

[13] BENTES, L. M. A. “**Sistema de segurança veicular com uso de GPS baseado em Arduino**”, Graduação. Amazonas: Universidade do Estado do Amazonas, Escola Superior de Tecnologia, 2013.

[14] BESZCZYNSKI, Leandro. “**Protótipo de um sistema de rastreamento veicular baseado no módulo TELIT**”, Graduação. Santa Catarina: Universidade Regional de Blumenau, Centro de Ciências Exatas e Naturais, 2008.

[15] GUARAGNI, Vinícius. “**Sistema de monitoramento no transporte de resinas em caminhão tanque**”. Graduação. Rio Grande do Sul: Universidade FEEVALE, 2014.

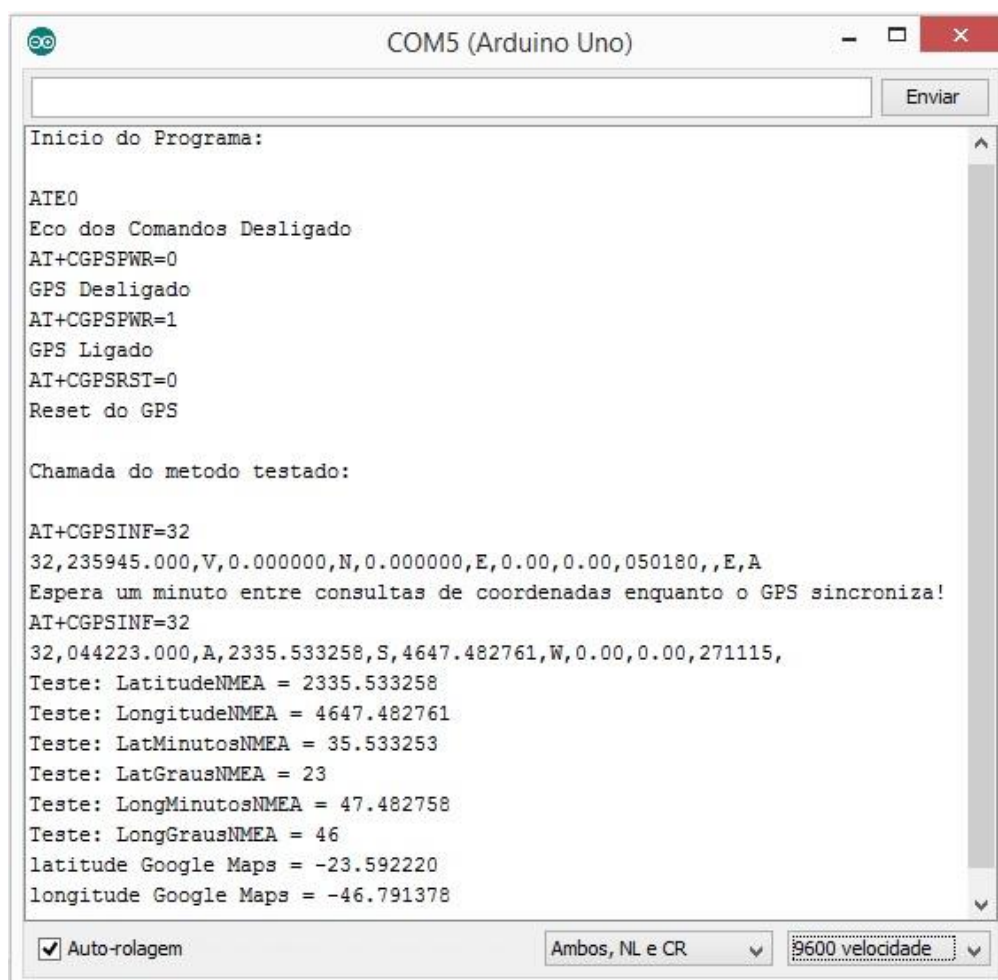
[16] Brasil. **Código Penal**. Disponível em:  
[http://www.planalto.gov.br/ccivil\\_03/decreto-lei/Del2848compilado.htm](http://www.planalto.gov.br/ccivil_03/decreto-lei/Del2848compilado.htm). Acessado em: 01/09/2015.

[17] Arduino, **Produtos**. Disponível em:  
<https://www.arduino.cc/en/Main/Products>. Acessado em: 01/09/2015



## APÊNDICE A – Testes com o SIM908

A figura 19 ilustra a obtenção do código NMEA através do GPS, utilizando a função *getLatLong()*, que neste teste é chamada a partir da função principal *void setup()* da IDE Arduino. No código final do projeto, a função *getLatLong()* é utilizada na função *void loop()* quando recebido SMS referente à localização do veículo:



```
Início do Programa:

ATE0
Eco dos Comandos Desligado
AT+CGPSPWR=0
GPS Desligado
AT+CGPSPWR=1
GPS Ligado
AT+CGPSRST=0
Reset do GPS

Chamada do metodo testado:

AT+CGPSINF=32
32,235945.000,V,0.000000,N,0.000000,E,0.00,0.00,050180,,E,A
Espera um minuto entre consultas de coordenadas enquanto o GPS sincroniza!
AT+CGPSINF=32
32,044223.000,A,2335.533258,S,4647.482761,W,0.00,0.00,271115,
Teste: LatitudeNMEA = 2335.533258
Teste: LongitudeNMEA = 4647.482761
Teste: LatMinutosNMEA = 35.533253
Teste: LatGrausNMEA = 23
Teste: LongMinutosNMEA = 47.482758
Teste: LongGrausNMEA = 46
latitude Google Maps = -23.592220
longitude Google Maps = -46.791378
```

**Figura 19** - Resultado da função *getLatLong()*

**Fonte:** O Autor.

A figura 20 fornece a estrutura do SMS obtido pelo módulo SIM908 através do comando **AT+CMGR=1**, e a figura 21 ilustra a comunicação do módulo SIM908

utilizando a tecnologia SMS, através da função `enviaSms()`, chamada dentro da função principal `void loop()` da IDE Arduino.

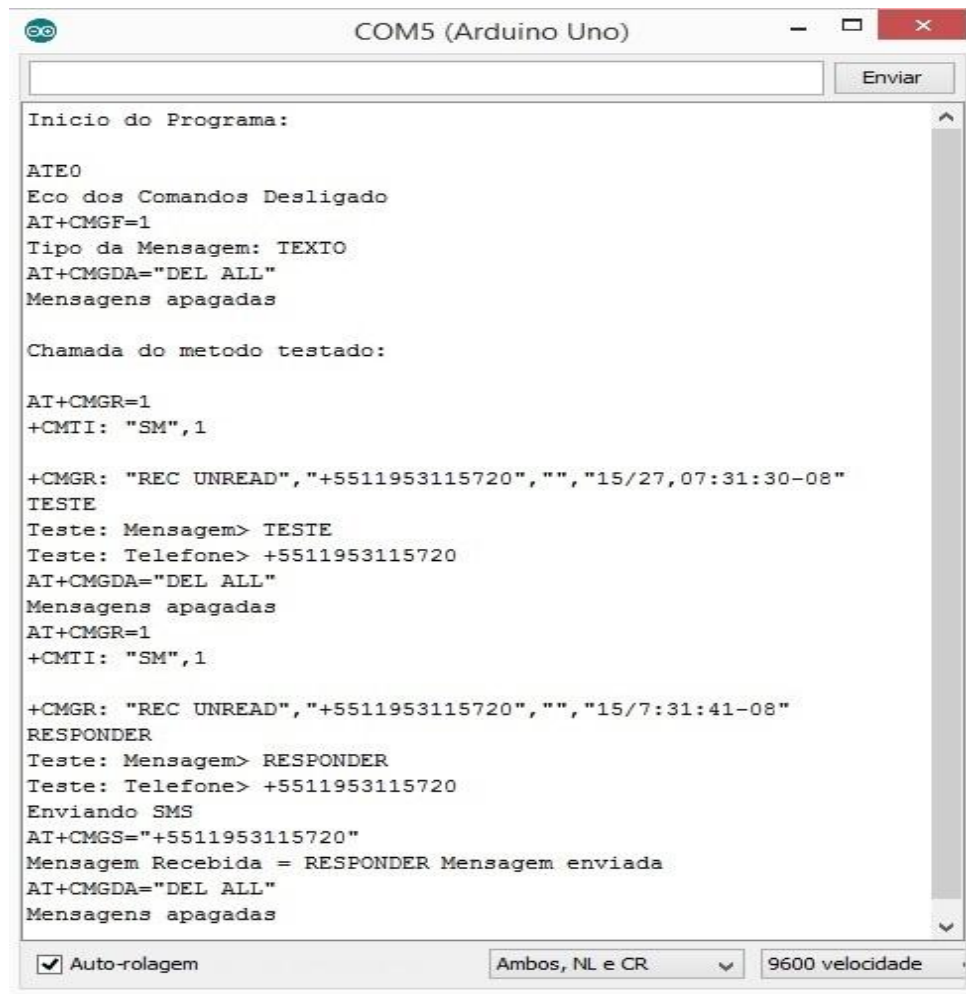
```
OK<EOL><CR><EOL><CR>+CMTI:<SPACE>"SM",1<EOL><CR><EOL><CR>+CMGR:<SPACE>"REC UNREAD",  
"telefone", "", "data,hora"<EOL><CR>mensagem<EOL><CR><EOL><CR>OK
```

**Figura 20** - Estrutura da Mensagem SMS

**Fonte:** O Autor

Através da estrutura apresentada é possível obter o conteúdo da mensagem recebida, em formato texto, obter o telefone do usuário, em formato internacional, para uso no sistema de cadastros e de privilégios e obter a data e hora em que a mensagem foi enviada, o que pode ser útil em aplicações que façam uso de banco de dados.

As mensagens recebidas são classificadas e armazenadas pelo módulo SIM908 como: *REC UNREAD*, mensagens recebidas ainda não lidas, *REC READ*, mensagens recebidas lidas, *STO UNSENT*, mensagens armazenadas não enviadas, *STO SENT*, armazenadas e enviadas e *ALL*, todas as mensagens. O comando **AT+CMGR=1**, utilizado no projeto, obtém o conteúdo de uma mensagem não lida, isto é, que acabou de chegar ao sistema, e muda automaticamente sua classificação para mensagem armazenada lida.



```
Inicio do Programa:

ATE0
Eco dos Comandos Desligado
AT+CMGF=1
Tipo da Mensagem: TEXTIO
AT+CMGDA="DEL ALL"
Mensagens apagadas

Chamada do metodo testado:

AT+CMGR=1
+CMTI: "SM",1

+CMGR: "REC UNREAD", "+5511953115720", "", "15/27, 07:31:30-08"
TESTE
Teste: Mensagem> TESTE
Teste: Telefone> +5511953115720
AT+CMGDA="DEL ALL"
Mensagens apagadas
AT+CMGR=1
+CMTI: "SM",1

+CMGR: "REC UNREAD", "+5511953115720", "", "15/7:31:41-08"
RESPONDER
Teste: Mensagem> RESPONDER
Teste: Telefone> +5511953115720
Enviando SMS
AT+CMGS="+5511953115720"
Mensagem Recebida = RESPONDER Mensagem enviada
AT+CMGDA="DEL ALL"
Mensagens apagadas
```

**Figura 21** - Resultado da função `enviaSms()`

**Fonte:** O Autor

A função `enviaSms()` só é chamada durante o teste quando a mensagem **“RESPONDER”** é enviada ao Arduino. A resposta enviada pelo módulo SIM908 pode ser observada na figura 22. Além do comando **AT+CMGR=1**, durante a função principal `void loop()` também é chamado o comando **AT+CMGDA=DELL ALL**, responsável por excluir todas as mensagens armazenadas no cartão SIM do *Shield*.



**Figura 22** - Resposta do SIM908 ao SMS Enviado

**Fonte:** O Autor

Os códigos utilizados durante os testes, assim como os códigos finais do hardware embarcado e da interface estão disponíveis dentro do CD entregue com a monografia, cuja estrutura é descrita no APÊNDICE E.

## APÊNDICE B – Arquivo Manifest.xml

O Arquivo *Manifest.xml* descreve a interface gráfica, as permissões que o usuário deve conceder para sua utilização e as classes responsáveis pela manipulação de cada elemento.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="epusp.projetoformatura" >
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission
android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
    <!--
    The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
    Google Maps Android API v2, but are recommended.
    -->
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <!--
Envio/Recebimento e Leitura/Escrita de SMS.
-->
    <uses-permission android:name="android.permission.RECEIVE_SMS"></uses-permission>
    <uses-permission android:name="android.permission.SEND_SMS"></uses-permission>
    <uses-permission android:name="android.permission.READ_SMS"></uses-permission>
    <uses-permission android:name="android.permission.WRITE_SMS"></uses-permission>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <meta-data
            android:name="com.google.android.gms.version"
            android:value="@integer/google_play_services_version" />
        <meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="@string/google_maps_key" />
        <activity
            android:name=".SMSSender"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver android:name=".SMSReceiver" android:enabled="true" android:exported="true"
android:permission="android.permission.BROADCAST_SMS">
            <intent-filter android:priority="1000">
                <action android:name="android.provider.Telephony.SMS_RECEIVED"/>
            </intent-filter>
        </receiver>
    </application>
</manifest>
```

## APÊNDICE C – Classe SMSReceiver

A classe *SMSReceiver* é responsável pela manipulação das mensagens enviadas pelo módulo SIM908 ao celular do usuário, e a atualização da latitude e da longitude do mapa quando pedido de localização do veículo é feito pelo usuário.

```
package epusp.projetoformatura;

import android.os.Bundle;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.telephony.SmsMessage;

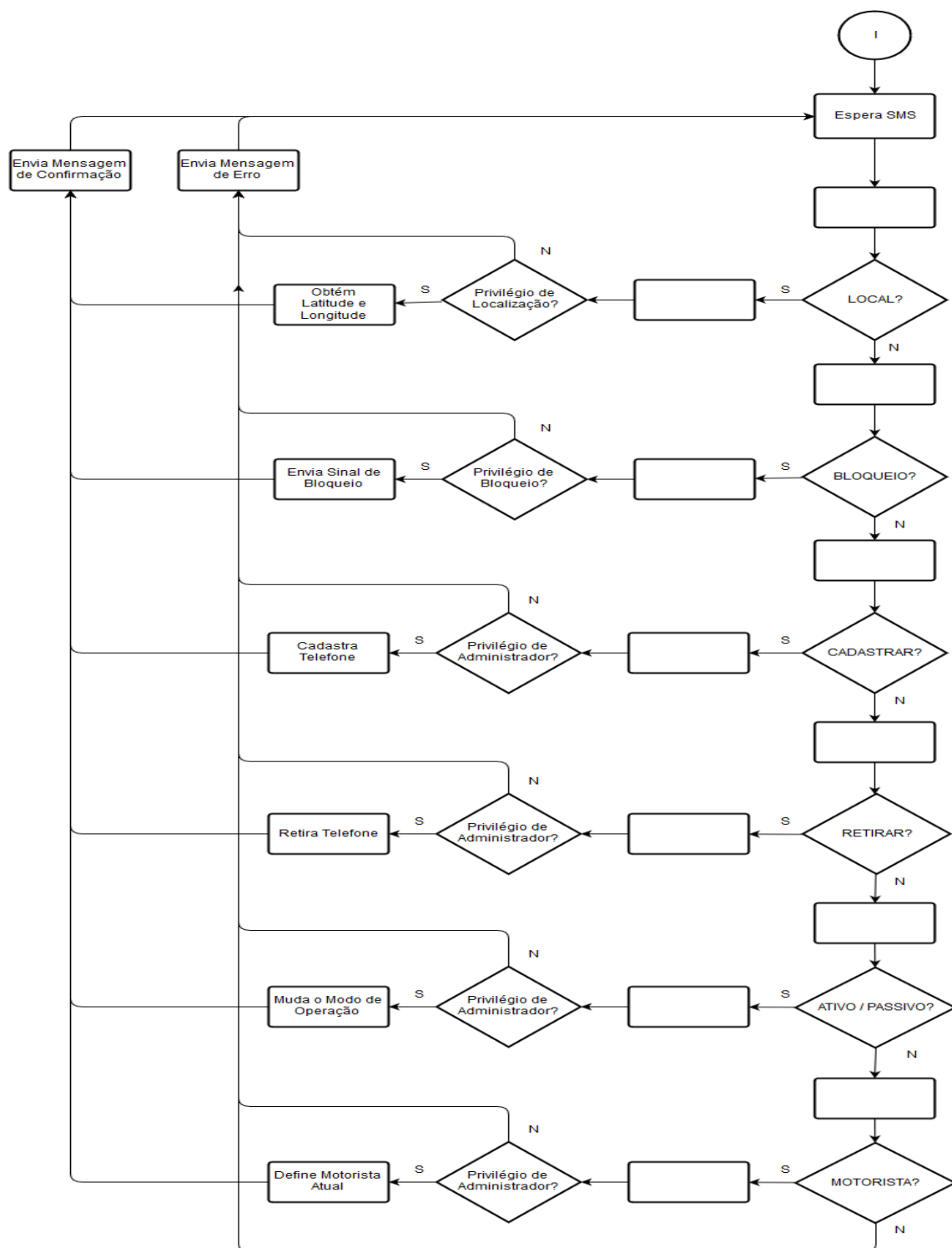
/**
 * Esta classe manipula os SMSs recebidos
 */
public class SMSReceiver extends BroadcastReceiver {

    private static final String TAG = "SMSReceiver";
    private static String latitude = "";
    private static String longitude = "";

    public void onReceive(Context context, Intent intent) {
        Bundle bundle = intent.getExtras();
        if (bundle != null) {
            Object[] pduObj = (Object[]) bundle.get("pdu");
            SmsMessage[] messages = new SmsMessage[pduObj.length];
            // Coleta as informações do SMS do Pdu.
            for (int i = 0; i < pduObj.length; i++) {
                messages[i] = SmsMessage.createFromPdu((byte[]) pduObj[i]);
            }
            for (SmsMessage currentMessage : messages) {
                if (SMSSender.numTelefone.contains( currentMessage.getDisplayOriginatingAddress() )){
                    SMSSender.setTexto(currentMessage.getDisplayMessageBody());
                    if(currentMessage.getMessageBody().startsWith("OK") &&
currentMessage.getMessageBody().length() > 2) {
                        int index = currentMessage.getMessageBody().lastIndexOf(':');
                        for (index = (index + 1); currentMessage.getMessageBody().charAt(index) != ',';
index++)

                            latitude = latitude + currentMessage.getMessageBody().charAt(index);
                            for (index = (index + 1); index < currentMessage.getMessageBody().length(); index++)
                                longitude = longitude + currentMessage.getMessageBody().charAt(index);
                            SMSSender.setLatLong(latitude, longitude);
                        }
                    }
                }
            }
        }
    }
}
```

## APÊNDICE D – Fluxograma de Mensagens



**Figura 23 - Fluxograma de Mensagens**

**Fonte:** O Autor

## APÊNDICE E – Estrutura do CD

O CD entregue em conjunto com a monografia é composto de três pastas principais: “Documentação”, “Testes” e “Projeto Final”.

A pasta “Documentação” possui a versão digital da monografia em formato “.pdf” e o *press release* e *banner* entregues anteriormente. Esta pasta também possui uma planilha com as tabelas traduzidas que foram utilizadas neste projeto e a pasta “Imagens”, com todas as imagens produzidas pelo autor deste documento.

A pasta “Testes” possui as pastas “TesteGPS” e “TesteGSM”, com os códigos com extensão “.ino”, formato utilizado pela IDE Arduino, utilizados nos testes de comunicação GSM e GPS descritos no Apêndice A.

A pasta “Projeto Final” possui a pasta Rastreamento\_Bloqueio\_Veiculos, com o código do hardware embarcado em um arquivo de extensão “.ino”, e a pasta “ProjetoFormatura”, com a interface gráfica, no formato utilizado pelo Android Studio.



**Figura 24:** Estrutura do CD

**Fonte:** O Autor



## ANEXO A – Diagrama Arduino Uno

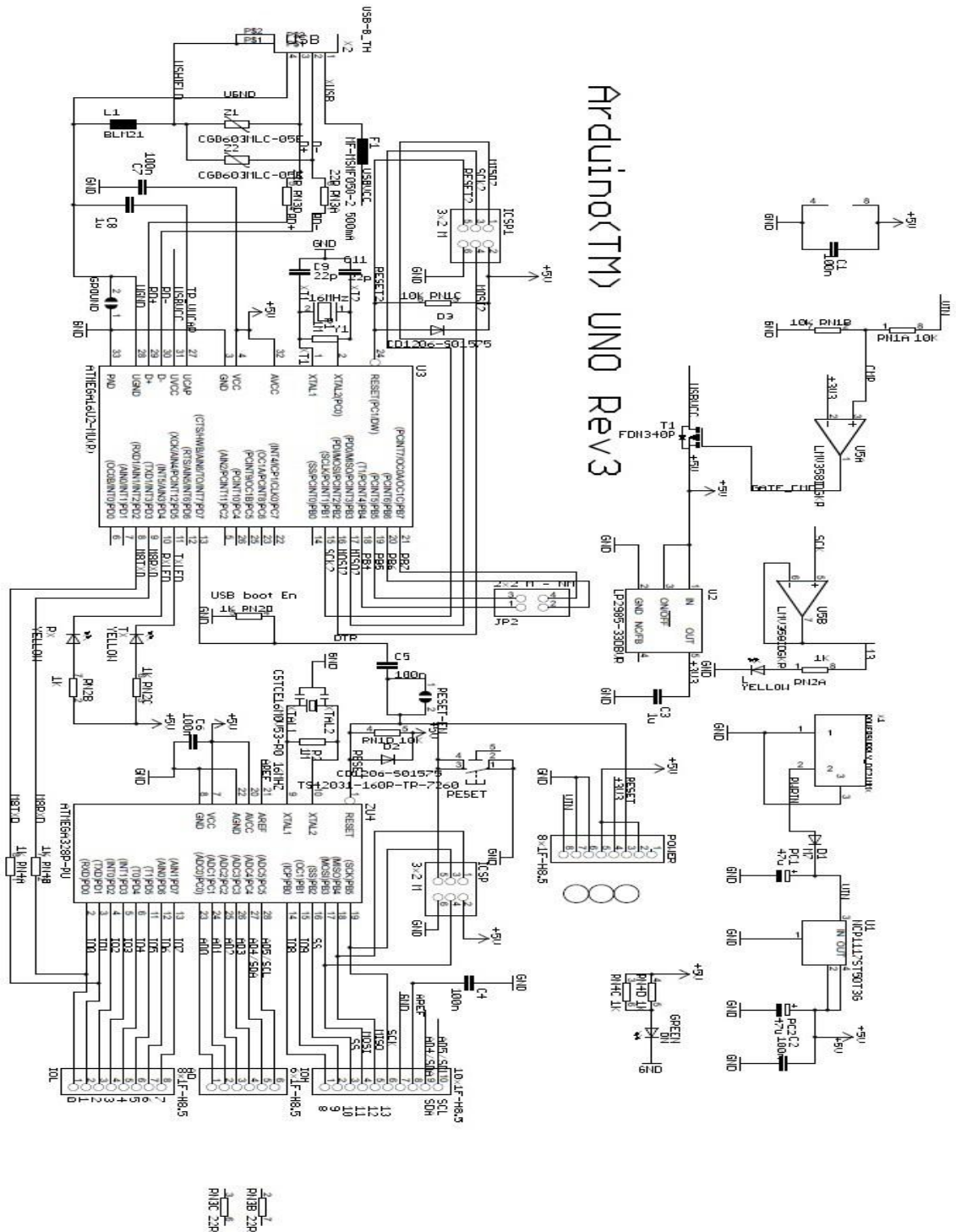


Figura 25 - Diagrama Arduino Uno

Fonte: [https://www.arduino.cc/en/uploads/Main/Arduino\\_Uno\\_Rev3-schematic.pdf](https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf)

## ANEXO B – Características Elétricas do SIM908

No quadro 10 são apresentados os valores máximos, mínimos e típicos da tensão de alimentação, tensão de entrada, corrente de entrada e corrente de saída.

Símbolo	Parâmetro	Min	Max	Unidade
V <sub>BAT</sub>	Tensão de Alimentação	-	5.5	V
V <sub>I</sub> *	Tensão de Entrada	-0.3	3.1	V
I <sub>I</sub> *	Corrente de Entrada	-	10	mA
I <sub>O</sub> *	Corrente de Saída	-	10	mA

**Quadro 10** - Níveis máximos absolutos

**Fonte:** Tradução de SIM908 Hardware Design, p. 43.

No quadro 11, são apresentadas as temperaturas de operação e armazenamento máximas, mínimas e típicas.

Símbolo	Parâmetro	Min	Típico	Max	Unidade
V <sub>BAT</sub>	Tensão de Alimentação	3.2	4.0	4.8	V
T <sub>OPER</sub>	Temperatura de Operação	-40	25	85	°C
T <sub>STG</sub>	Temperatura de Armazenamento	-45		90	°C

**Quadro 11** - Condições Recomendadas de Operação

**Fonte:** Tradução de SIM908 Hardware Design, p. 43.