# Monitoramento e travamento de dispositivos via satélite

**Lucas Flores Machado[1], Fernando Sarturi Prass[1]**

[1]Ciência da Computação – Centro Universitário Franciscano – (UNIFRA)

Rua dos Andradas, 1614 – 97.010-032 – Santa Maria – RS – Brasil

lucas.fmachado19@gmail.com, fernando.prass@unifra.br

***Abstract.*** *Faced with the amount of goods transported daily by land vehicles, this work has the motivation to increase reliability and safety in these deliveries. The project consists of the development of a system for opening and locking compartments of such vehicles, tracking them through their current geographical position. Thus, an Arduino board integrated to a GPS Shield will be used to ensure that the latch is released only when the vehicle is within a minimum radius of delivery places previously selected by the user through Google Maps.*

***Resumo.*** *Frente à quantidade de mercadorias transportadas diariamente por veículos terrestres, este trabalho tem como motivação aumentar a confiabilidade e a segurança nestas entregas. O projeto consiste no desenvolvimento de um sistema de abertura e travamento do compartimento destes veículos, monitorando-os através da posição geográfica atual em que se encontram. Desta forma, será utilizada uma placa Arduino integrada a um Shield GPS, para garantir que a trava só seja liberada quando o veículo se encontrar dentro de um raio de distância mínima dos locais de entregas selecionados previamente pelo usuário através do Google Maps.*

## 1. Introdução

Segundo dados estatísticos dos Correios (2015), em 2014 foram distribuídos cerca de 8,5 bilhões de mercadorias, com previsão de aumento para os próximos anos. Para conseguir executar todas as demandas, a empresa conta com uma frota de 24.681 veículos em todo o território brasileiro. E esta é apenas uma das empresas do mercado, há outras, como Braspress, GolLog e FedEx.

Apesar do controle das entregas evoluir constantemente devido a aperfeiçoamentos dos veículos ou dos sistemas que auxiliam o serviço, um volume muito grande de mercadorias pode resultar em extravio ou perda de itens.

Os compartimentos da maioria dos veículos em que são armazenados os objetos são trancados por meio de uma trava de segurança manual ou até mesmo liberados através de um botão na cabine do motorista, o que permite que qualquer pessoa possa destrancá-los em qualquer lugar, mesmo o veículo não estando em um dos locais de entrega destinados.

Visando oferecer uma alternativa mais segura e que possa garantir mais integridade aos produtos transportados, o presente trabalho tem como objetivo

desenvolver uma solução que garanta a abertura e travamento dos compartimentos de cargas dos veículos por meio de sua localização geográfica.

Desta forma, tem-se a garantia de que o veículo só possa ter o compartimento destravado quando o mesmo estiver num raio mínimo de um dos locais de entrega, determinados previamente pelo usuário.

Para o desenvolvimento deste trabalho, foi utilizada uma placa de Arduino integrada a um Shield GPS, possibilitando, assim, controlar os locais de entrega e calcular as coordenadas dos mesmos através do Google Maps.

## 2. Referencial Teórico

Esta seção apresenta as especificações das tecnologias e conteúdos utilizados para desenvolver o projeto.

### 2.1 Microcontroladores

Um microcontrolador é constituído de um microprocessador, memória e periféricos de entrada/saída e pode ser programado para funções específicas, como, por exemplo, o controle de máquinas e diferentes automações. Por isso, geralmente operam em uma frequência muito baixa, porém adequada à maioria das aplicações para as quais são designados [Bentes 2013; Cavalcante et al. 2011].

Este trabalho fez uso de um microcontrolador ATMega 2560, que é utilizado pelos dispositivos Arduino Mega 2560 Rev3. Ele é um chip fabricado pela empresa Atmel, com arquitetura RISC (*Reduced Instruction Set Computer*), que se refere a um conjunto reduzido de instruções com curto período de tempo para serem executadas [Sram and Cycles 2014].

### 2.2 Plataforma Arduino

Arduino é uma plataforma de prototipagem eletrônica baseado em *hardware* e *software* livres, flexíveis e fáceis de usar. É destinado a todos aqueles que estão interessados em criar projetos ou ambientes interativos [Lima et al. 2014].

A placa de Arduino permite que se possa interagir com o ambiente externo recebendo em suas entradas sinais provenientes de sensores como, por exemplo, temperatura e pressão. Com os valores destas entradas é possível processar os dados recebidos em um microcontrolador e disponibilizar os valores resultantes de volta ao ambiente por meio do acionamento de atuadores [Banzi 2009; Lima et al. 2014].

Segundo Cavalcante et al. (2011) e Ferreira (2005), como o Arduino é uma plataforma *open source*, ou seja, criada para o domínio público, pode ser modificada e aprimorada por outras pessoas conforme necessidades. Por isso, a plataforma se tornou muito utilizada, tanto para se desenvolver objetos interativos autônomos como para ser integrada a um *software* de computador.

O modelo da placa de Arduino usado neste projeto foi o Arduino Mega 2650 Rev3. Este modelo possui um *chip* do tipo ATmega2560 (abordado no tópico 2.1 Microcontroladores) [Arduino 2015].

O Arduino possui uma linguagem de programação própria, baseada em Wiring (um subconjunto das linguagens C e C++), a qual é implementada em um ambiente de desenvolvimento (IDE) baseado em Processing, e que pode ser utilizada em vários sistemas operacionais, como Windows, Linux e Mac [Gioppo et al. 2009].

### 2.3 Shield GPS (Sistema de Posicionamento Global)

É possível inserir funcionalidades extras ao Arduino conectando placas adicionais chamadas Shields, cuja funcionalidade é expandir sua capacidade. Exemplo: podem ser utilizados Shields para prover comunicação Ethernet, Bluetooth, Wifi e até mesmo controladores GPS (o qual foi utilizado neste projeto) [Lima et al. 2014].

Os Shields são adaptadores feitos para serem utilizados juntamente com as placas de Arduino, de modo a serem adicionadas funcionalidades a eles. São placas *plug-and-play*, com conectores montados para serem acoplados diretamente ao Arduino.

Quando conectados, os Shields estendem os seus pinos de entrada/saída até o topo de suas próprias placas de circuito para que o usuário continue tendo acesso a todos eles mesmo depois de conectados.

Neste projeto foi adicionada um Shield de GPS do modelo ITEAD Studio, este Shield é baseado no módulo GPS u-box NEO-6M e contém múltiplos receptores, permitindo assim um posicionamento preciso e sendo útil para várias aplicações, como navegação automotiva, posição pessoal e navegação marítima [Arnizaut 2014].

### 2.4 Google Maps

O Google Maps é um serviço do Google que oferece uma tecnologia de mapas amigáveis e informações locais, incluindo a localização, informações de contatos e direções de condução [Gibson and Schuyler 2006].

Com o passar do tempo, foram sendo adicionadas novas funcionalidades para os usuários, gerando comodidade e facilidades que não haviam sido fornecidas até então por nenhum outro serviço do tipo. Com ele é possível que se faça desde cálculo de rotas, visualizações 3D de ruas e edificações, até fornecimento de informações sobre tráfego e sobre o transporte público.

O Google Maps tem como funcionalidade principal a exibição de um mapa *website*, partindo de uma coordenada que é exibida centralizada na tela. Isso já basta para que o usuário localize ruas e regiões aos arredores do endereço fornecido.

Segundo Purvis et al. (2006), com o grande sucesso e aceitação dos usuários ao serviço Google Maps, pouco tempo depois foi lançada a sua API (Application Programming Interface), a qual foi utilizada neste projeto. Ela permite aos usuários inserir mapas em suas páginas *web*, contando com a possibilidade de personalização e customização dos mesmos de acordo com suas preferências.

### 2.5 GPS

Segundo Monico (2001), o GPS é um sistema de radionavegação desenvolvido pelo Departamento de Defesa (DoD) dos Estados Unidos, com o intuito de ser o principal sistema de navegação das forças armadas americanas.

O autor ainda afirma que, em razão da alta acurácia proporcionada pelo sistema e do grande desenvolvimento da tecnologia envolvida nos receptores GPS, uma grande comunidade usuária emergiu dos mais variados segmentos da comunidade civil (navegação, posicionamento geodésico, agricultura, controle de frotas, etc.).

A concepção do sistema GPS permite que um usuário, em qualquer local da superfície terrestre ou próximo a ela, tenha a sua disposição, no mínimo, quatro satélites para serem rastreados.

No GPS há dois tipos de serviços, conhecidos como SPS (Serviço de Posicionamento Padrão) e PPS (Serviço de Posicionamento Preciso). O primeiro é um serviço de posicionamento e tempo padrão que está disponível a todos os usuários do globo. Já o segundo proporciona ao usuário melhores resultados, mas é restrito somente ao uso militar e a usuários autorizados [Monico 2001].

### 2.5.1 Principais Segmentos do GPS

Monico (2001) classifica o GPS em três segmentos principais: espacial, de controle e de usuários. Segue a especificação:

Segmento espacial: consiste em 24 satélites distribuídos em seis planos orbitais igualmente espaçados, com quatro satélites em cada plano numa altitude aproximada de 20.200km. Os planos orbitais são inclinados 55º em relação ao Equador, e o período orbital é de aproximadamente 12 horas siderais. Desta forma, a posição dos satélites se repete, a cada dia, 4 minutos antes que a do dia anterior.

Na Figura 1 pode-se observar a constelação dos satélites:

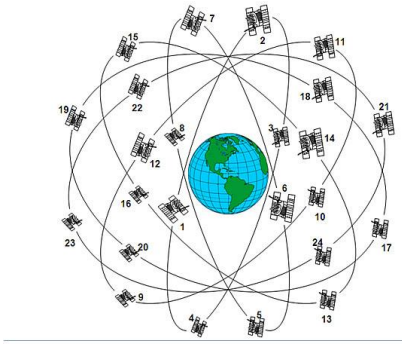

**Figura 1. Segmentação espacial [Da bússola ao GPS 2011]**

Segmento de controle: possui tarefas principais a serem desenvolvidas, são elas: (1) Monitorar e controlar continuamente o sistema de satélite; (2) Determinar o sistema de tempo GPS; (3) Calcular as correções dos relógios dos satélites; e (4) Atualizar periodicamente as mensagens de navegação de cada satélite.

O sistema de controle é composto por cinco estações monitoras (Hawaii, Kwajalein, Ascension Island, Diego Garcia, Colorado Springs). Três delas possuem antenas que transmitem os dados para os satélites (Ascension Island, Diego Garcia, Kwajalein), e uma estação de controle central localizada em Colorado Springs, como pode ser observado na Figura 2:


**Figura 2. Segmento de controle [Monico 2001]**

Segmento de usuários: é constituído pelos receptores GPS, os quais devem ser apropriados para os propósitos a que se destinam. As categorias de usuários podem ser divididas em civil e militar, em que os militares fazem uso dos receptores GPS para estimar suas posições e deslocamento quando realizam manobras de combate e de treinamento. Já para o uso de usuários civis, há uma grande quantidade de receptores no mercado para diversas aplicações [Monico 2001].

## 2.6 Sistema de Coordenadas

O usuário de Sistema de Informação Geográfica (SIG, que será abordado na seção 2.7 Sistema de Informação Geográfica (SIG)) está acostumado a navegar em seus dados através de ferramentas simples como apontamento na tela com o cursor e a subsequente exibição das coordenadas geográficas da posição indicada. Porém, por trás da simplicidade aparente dessa ação, há algumas transformações entre diferentes sistemas de coordenadas que garantem a relação entre um ponto na tela do computador e as coordenadas geográficas apresentadas ao usuário [D'Alge 1991].

O autor ainda cita os cinco tipos de sistemas de coordenadas existentes, como:

- Sistemas de Coordenadas Geográficas;
- Sistema Geocêntrico Terrestre;
- Sistema de Coordenadas Planas;
- Sistema de Coordenadas Polares; e
- Sistema de Coordenadas de Imagem.

## 2.7 Sistema de Informação Geográfica (SIG)

O termo SIG é aplicado para sistemas que realizam o tratamento computacional de dados geográficos e recuperam informações não apenas com base em suas características alfanuméricas, mas também através de sua localização espacial; para que isto seja possível, a geometria e os atributos dos dados em um SIG devem estar georreferenciados, isto é, localizados na superfície terrestre e representados em uma projeção cartográfica [Davis and Neto 2001; Druk et al. 2004; Monico 2001].

Monico (2001) afirma que, num enfoque orientado a processos, o SIG pode ser definido como coleções de subsistemas integrados, em que dados espaciais passam por uma sequência de processos de conversão, coleta, armazenamento e manipulação.

Em uma visão abrangente a respeito destes processos, Druk et al. (2004) indica os principais componentes de um SIG. São eles:

- Interface com usuário;
- Entrada e edição de dados;
- Funções de processamento gráfico e de imagens;
- Visualização e plotagem; e
- Armazenamento e recuperação de dados (organizados sob forma de um banco de dados geográficos).

Ainda, segundo o autor, estes componentes se relacionam de forma hierárquica. A interface homem máquina define como o sistema é operado e controlado. No nível intermediário é onde se deve ter os mecanismos de processamento de dados espaciais já citados (por exemplo entrada, edição, análise, visualização e saída). Internamente ao sistema, um banco de dados geográficos armazena a recupera os dados espaciais. Cada sistema, em função de seus objetivos e necessidades, implementa esses componentes de forma distinta, porém todos os subsistemas citados formam um SIG.

Basicamente, o SIG pode ser visto como uma tecnologia computacional que possibilita, através de computadores e *softwares*, a análise de informações dentro de um contexto geográfico. Os dados que são produzidos em um SIG são os mais variados e dependem do propósito a que se destinam. Pode-se ainda, dentre outras possibilidades, apresentar uma definição segundo sua aplicação, tal como um sistema para apoio à tomada de decisões ou para análise de dados geográficos. Por exemplo, um SIG destinado à educação interessa-se não somente por dados como idade, infraestrutura e segurança, mas também pela localização (posição) de alguns, ou todos, os dados envolvidos. Isso é possível, pois os dados que produzem um SIG permitem integrar dados coletados em diferentes instantes e escalas e usando diferentes métodos de aquisição [Druk et al. 2004; Santos and Pena 2011].

Ainda de acordo com os autores, há pelo menos três grandes maneiras de utilizar um SIG:

- Como ferramenta para produção de mapas;
- Como suporte para análise espacial de fenômenos; e
- Como um banco de dados geográficos (para funções de armazenamento e recuperação de informação espacial).

## 2.8 Trabalhos Correlatos

No trabalho **"Sistema GPS Android"** de Pappis e Prass (2012) é apresentado um sistema de rastreamento de veículos composto por um *software* de rastreamento móvel, utilizando aparelhos celulares comuns com o sistema operacional Android e de um sistema *web* que recolhe as informações obtidas pelo celular.

O trabalho citado anteriormente possui similaridades com este projeto, pois também utiliza a API do Google Maps e geolocalização. A diferença é que, no trabalho citado, as informações são utilizadas para o rastreamento, monitoramento e controle de rotas de veículos que utilizam o sistema desenvolvido.

Já no trabalho **"Sistema de segurança veicular com uso de GPS baseado em Arduino"** de Bentes (2013) é desenvolvido um sistema voltado a segurança veicular, que permite o rastreamento de veículos em tempo real. Além da função de rastreamento, também é possível desligar o veículo remotamente por meio de uma aplicação *web*.

Algumas similaridades foram encontradas, como, por exemplo, o uso de uma placa Arduino integrada a um Shield GPS, e a manipulação de uma API do Google Maps para que fosse possível fazer o rastreamento e monitoramento constante do veículo através da localização geográfica do mesmo. Porém, a diferença encontrada é que, no trabalho citado, os usuários fazem uso de uma aplicação *web* para gerenciar o veículo.

## 3. Metodologia

Para o desenvolvimento deste trabalho, foi utilizada a metodologia *Feature Driven Development* (Desenvolvimento Guiado por Funcionalidades – FDD) por se adaptar melhor aos objetivos requeridos pelo projeto.

Segundo Palmer e Felsing (2002), a metodologia FDD é construída a partir de um conjunto de "melhores práticas". Com isso, é possível que se faça uso somente de algumas das práticas englobadas no processo, não se alcançando, dessa forma, todos os benefícios que se pode ter utilizando a metodologia completa.

O FDD é classificado por Retamal (2014) como sendo uma metodologia objetiva e estruturada. No artigo "*Feature Driven Development* – Descrição dos Processos" o autor separa a metodologia FDD em cinco processos, sendo eles:

1. DMA (Desenvolver um modelo abrangente): Realiza-se um estudo dirigido sobre o escopo do sistema e seu contexto;
2. CLF (Construir uma lista de funcionalidades): Realiza uma atividade para identificar todas as funcionalidades que satisfaçam os requisitos;
3. PPF (Planejar por funcionalidade): Planeja a ordem na qual as funcionalidades serão implementadas para se produzir o projeto;
4. DPF (Detalhar por funcionalidade): Trata-se de uma atividade para cada funcionalidade do sistema, e tem como objetivo produzir um pacote de projeto para a funcionalidade;
5. CPF (Construir por funcionalidade): Faz uso do pacote de projeto, realizado na etapa anterior para que seja implementado os itens necessários para que suas classes suportem o projeto para esta funcionalidade.

Na primeira etapa do trabalho foram executados os quatro primeiros processos do FDD, elaborando um modelo de projeto, coletando e detalhando os requisitos necessários para o funcionamento do mesmo.

Nesta segunda etapa do trabalho contempla-se o quinto processo, onde é relacionado à construção do projeto com base nos requisitos levantados anteriormente.

# 4. Projeto

Esta seção detalha o funcionamento do projeto, em que são apresentadas as funcionalidades do mesmo como um todo.

## 4.1 Funcionalidades do sistema

Esta seção lista de forma resumida as funcionalidades do sistema, contemplados no processo 1 do FDD (DMA):

- Funcionalidade 1: Exibir mapa;
- Funcionalidade 2: Selecionar pontos de entrega;
- Funcionalidade 3: Escolher raio mínimo, a partir do ponto selecionado;
- Funcionalidade 4: Calcular coordenadas;
- Funcionalidade 5: Gerar arquivo texto com todas as possíveis coordenadas;
- Funcionalidade 6: Comparar coordenadas;
- Funcionalidade 7: Liberar trava.

## 4.2 Requisitos do sistema

Neste processo, como abordado na segunda etapa da metodologia (CLF), foi realizado um levantamento dos requisitos do sistema, separando-os em requisitos funcionais e requisitos não funcionais.

### 4.2.1 Requisitos funcionais

Requisitos funcionais estabelecem funcionalidades, características ou ações que o sistema deve fornecer. Basicamente, descrevem seu comportamento em determinadas situações e ações do usuário. Os principais requisitos funcionais deste projeto são:

- Exibir mapa ao usuário;
- Permitir que o usuário selecione os pontos de entrega desejados;
- Permitir que o usuário selecione o raio de distância desejado para cada ponto;
- Calcular todas as possíveis coordenadas para cada ponto selecionado pelo usuário;
- Comparar todas as coordenadas gravadas com a coordenada atual do veículo;
- Liberar a trava de segurança caso a coordenada atual coincida com alguma das coordenadas gravadas.

### 4.2.2 Requisitos não funcionais

Requisitos não funcionais estão relacionados a como será efetuado o uso do sistema. Geralmente, não são especificados pelo cliente, pois englobam as características

mínimas para o desempenho, manutenção e outros aspectos do sistema. Os principais requisitos não funcionais do sistema são:

- O sistema deve gravar todas as coordenadas em um arquivo texto;
- O Arduino deve fazer a comparação das coordenadas através do arquivo gravado no cartão SD.

### 4.3 Diagrama de Atividades

Um diagrama de atividade é essencialmente um gráfico que mostra o fluxo de controle de uma atividade para outra. Os diagramas de atividades são empregados para fazer a modelagem de aspectos dinâmicos do sistema. Na maior parte, isso envolve a modelagem das etapas sequenciais em um processo computacional [Booch et al. 2000].

Ainda seguindo a metodologia utilizada no projeto, a Figura 3 envolve a etapa 3 da metodologia (PPF), em que pode-se observar quais as atividades serão executadas para que o projeto possua o funcionamento correto, obedecendo a todas as etapas, explicadas a seguir.

Basicamente, o funcionamento do sistema se dá da seguinte forma: ao executar o sistema, o mapa é exibido para o usuário através do Google Maps para que seja selecionado um ponto de entrega e o raio de distância desejado para ele. Se o usuário desejar, pode repetir o processo quantas vezes quiser.

Depois de selecionados os pontos, o sistema calcula todas as possíveis coordenadas que se encontram dentro do raio escolhido pelo usuário para cada ponto de entrega, e grava-as. Com as coordenadas salvas, o Arduino faz uma constante comparação destas com a coordenada atual. Se a coordenada atual for igual a uma das coordenadas salvas, o Arduino permite a liberação da trava.

Na Figura 3 pode-se verificar o diagrama de atividades do projeto completo.

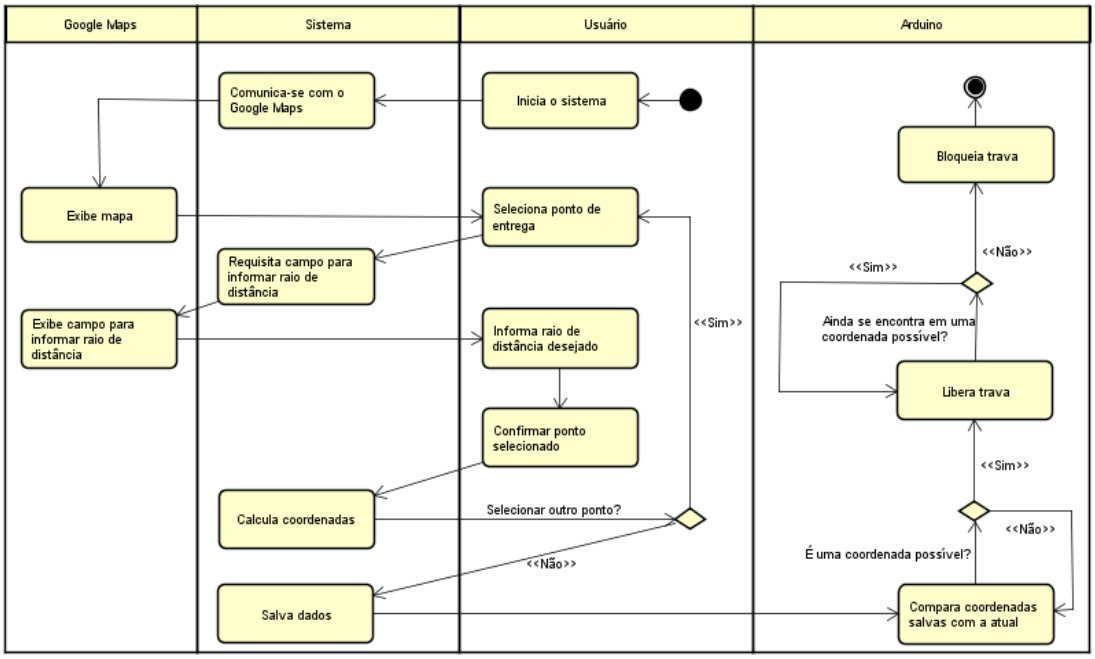


**Figura 3. Diagrama de Atividades**

### 4.4 Funcionamento do projeto

Este projeto integra a API do Google Maps com algumas funcionalidades do Arduino através do uso de Shields. Pondo em prática, desta forma, os conhecimentos obtidos no decorrer do curso.

Na Figura 4, pode-se observar, em um contexto geral, como foi elaborada a integração do projeto, em que todas as coordenadas mapeadas no Google Maps são gravadas em um arquivo texto em um cartão SD, que por sua vez é carregado no Arduino para que possa ser comparado com as coordenadas do veículo constantemente em tempo de execução do sistema.


**Figura 4. Integração do projeto**

Detalhadamente, o protótipo desenvolvido permite que o usuário selecione pontos desejados para efetuar entregas e, junto deles, também escolha um raio de distância para cada ponto selecionado, sendo este limitado entre um e cem metros. Para cada ponto escolhido, o sistema irá calcular e armazenar as coordenadas que permitem que a trava de segurança seja liberada, armazenando-as em um vetor, em que cada posição é armazenada no formato “latitude, longitude”.

Após selecionar todos os pontos, o usuário poderá selecionar a opção “Gerar Arquivo de Rotas”, onde será gravado um arquivo com todas as coordenadas dos pontos selecionados para que o usuário efetue o *download* do mesmo.

Após ser copiado para o cartão SD do Arduino, o arquivo será lido constantemente para que sejam comparadas as coordenadas atuais (obtidas pelo Shield GPS) com as coordenadas previamente salvas no arquivo. Caso as coordenadas coincidam, a trava será liberada até que as coordenadas atuais não sejam mais a mesma de nenhuma das coordenadas salvas no arquivo.

Na Figura 5, pode-se verificar um resumo do funcionamento do projeto. No bloco 1, vê-se o veículo a caminho do ponto de entrega, mas a trava se encontra bloqueada porque sua coordenada atual é diferente de todas as coordenadas previamente mapeadas no arquivo texto.

No bloco 2, a trava é liberada, uma vez que o veículo se encontra dentro da área de entrega. Isso acontece porque, ao ser feita a verificação do arquivo, pelo menos uma das coordenadas é igual à coordenada atual do veículo.

Por fim, no bloco 3, quando o veículo sai da área de entrega, é verificado novamente que no arquivo de texto não existe nenhuma coordenada igual a coordenada atual do veículo. E por esta razão a trava é bloqueada novamente.

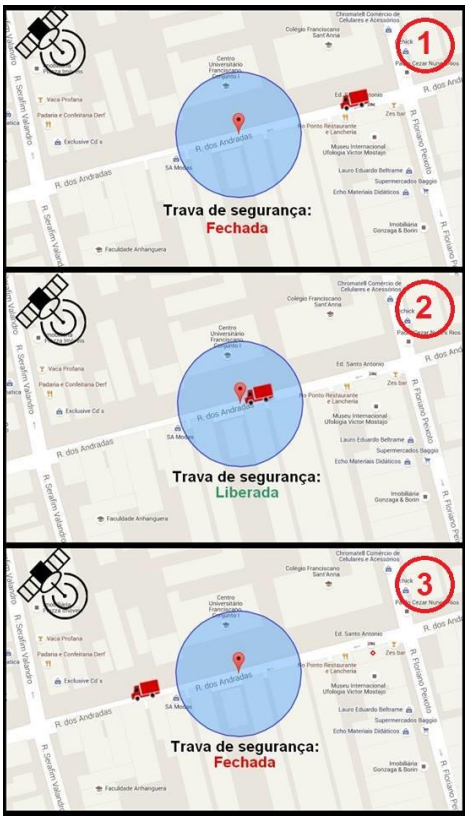


**Figura 5. Funcionamento do projeto**

## 4.5 Desenvolvimento

A Figura 6 apresenta o protótipo elaborado neste trabalho, seguido de uma breve explicação sobre cada tecnologia utilizada para implementação do protótipo.

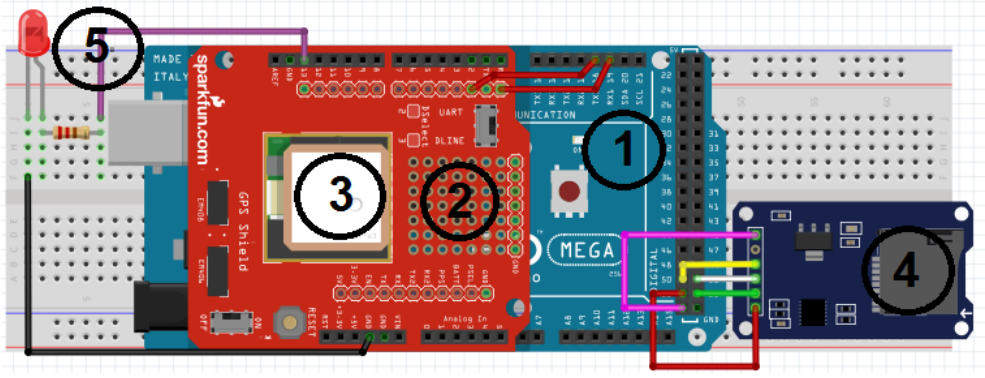


**Figura 6. Protótipo**

Na Figura 6, pode-se verificar os itens que foram utilizados para implementar o protótipo: (1) a placa de Arduino Mega; (2) o Shield GPS (ligado ao Arduino através dos pinos TX (pino zero) e RX (pino 1); (3) a antena de GPS, responsável por captar as coordenadas atuais, ligada diretamente ao Shield GPS; (4) o cartão SD, ligado no pino 53; e, (5) ligado ao pino 13, o LED que simboliza, neste caso, a trava de segurança do sistema.

Durante o desenvolvimento foram encontradas algumas dificuldades, que fizeram com que fossem alteradas ideias iniciais a respeito do projeto. As principais dificuldades encontradas, bem como a solução elaborada para resolvê-las, são:

(1) os dados lidos pelo Arduino no cartão SD eram convertidos em valores equivalentes em ASCII, fazendo com que cada coordenada tivesse um código difícil de ser identificado.

Para resolver o problema foi preciso fazer a conversão dos dados para cada caractere lido do arquivo texto fazendo, no final de cada linha, a comparação com a posição atual.

(2) não foi encontrada nenhuma função que busque automaticamente todas as coordenadas contidas dentro do círculo selecionado pelo usuário.

Para que pudessem ser mapeadas todas as coordenadas dentro do círculo selecionado foi preciso calcular os extremos do círculo, armazenando-os nas variáveis “coordCima”, “coordBaixo”, “coordEsquerda” e “coordDireita”, como pode ser verificado na Figura 7, em que é mostrado como foi desenvolvido o processo:

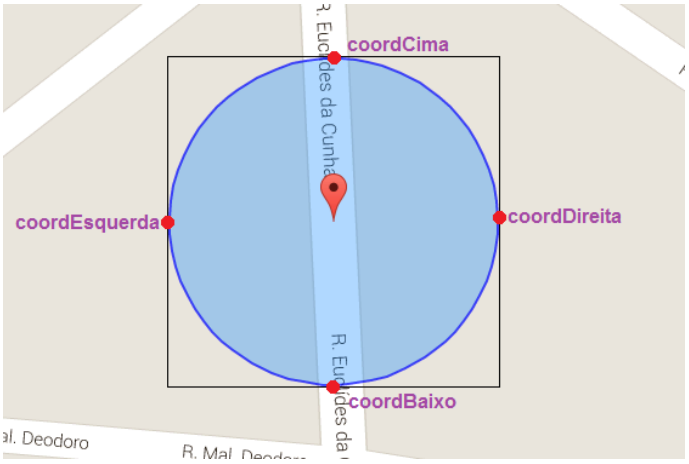


**Figura 7. Elaboração da matriz de coordenadas**

Com os pontos já armazenados nas variáveis, foi elaborada uma matriz, tomando como base as extremidades do círculo (como já ilustrado na Figura 7). A matriz é percorrida entre as coordenadas (em que as variáveis “i” e “j” simbolizam latitude e longitude, respectivamente), verificando a cada nova posição se a coordenada está contida dentro do círculo selecionado, como pode ser verificado no trecho de código da Figura 8.

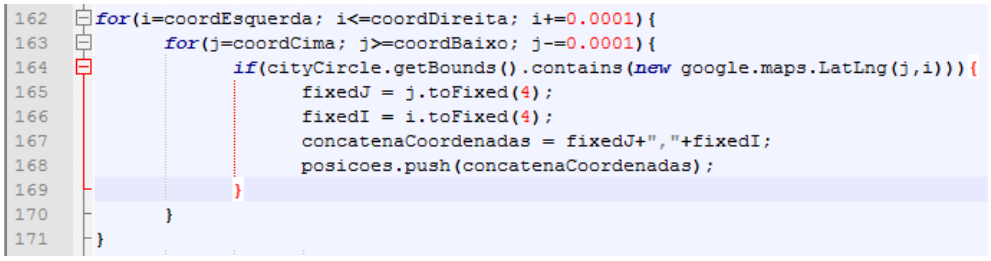


**Figura 8. Laço de repetição para percorrer a matriz de coordenadas**

Nas linhas 162 e 163 a matriz está sendo percorrida, limitando a longitude (que seriam as colunas da matriz) entre as extremidades "coordEsquerda" e "coordDireita" e a latitude (que seriam as linhas da matriz) entre as extremidades "coordBaixo" e "coordCima". Os pontos são percorridos a cada coordenada com uma precisão de 0,0001.

Na linha 164 é verificado se o ponto atual da matriz está contido no círculo selecionado pelo usuário. Se sim, as coordenadas serão fixadas em quatro casas decimais (linhas 165 e 166) e, em seguida, concatenadas no formato "latitude, longitude" (linha 167) para que, por fim, na linha 168 possam ser armazenadas no vetor de posições.

(3) as coordenadas dos pontos demoravam em ser calculadas para todos os pontos escolhidos.

Para isso, foi preciso limitar em quatro casas decimais todas as latitudes e longitudes lidas, diminuindo assim o número de coordenadas gravadas e, consequentemente, o tempo para leitura e verificação do arquivo.

(4) não é possível transferir um vetor de valores inteiro do JavaScript para o PHP.

Foi preciso concatenar todos os valores do vetor em uma *string*, demarcando por um caractere especial onde seriam as separações das posições do vetor. Para executar este procedimento foi preciso utilizar o método *join* do JavaScript e o *explode* do PHP, para transformar novamente a *string* em um vetor de coordenadas.

(5) Da forma que estava implementado anteriormente, o Arduino precisaria percorrer todo o arquivo para certificar-se de que a havia uma coordenada para liberação da trava. Desta forma, demorava para que fossem verificadas todas as coordenadas do arquivo, fazendo com que fosse preciso ficar por algum tempo em um determinado local para que a trava fosse liberada.

O código do Arduino foi modificado, fazendo com que não fosse mais preciso percorrer todo o arquivo para liberar a trava. Assim, no momento em que a coordenada atual coincidisse com uma das coordenadas do arquivo, a trava já é liberada no mesmo instante, não precisando percorrer o arquivo até o final. A trava permanece aberta enquanto o veículo ainda estiver dentro do raio de distância.

(6) Em locais fechados, algumas vezes, os testes apresentaram problemas de geolocalização, devido à baixa qualidade do Shield GPS utilizado.

Para resolver este problema, aconselha-se o uso de um Shield GPS de melhor qualidade e precisão, como os receptores GPS modelos GT-3731 e LS-20031. Destaca-se que eles não foram utilizados neste projeto por questões monetárias, já que aqui se deu maior atenção ao algoritmo de localização.

## 5. Conclusão

Durante o projeto foram pesquisados trabalhos que utilizaram as mesmas tecnologias em diferentes projetos. Estas pesquisas foram importantes para que fosse possível implementar e compreender determinadas funcionalidades, como, por exemplo, a interação do Shield GPS com o Arduino e a personalização do Google Maps através da API.

O protótipo se mostrou capaz não só de executar a verificação e a comparação das coordenadas, como também mostrou pleno funcionamento em testes práticos. Em certas ocasiões ocorrem alguns problemas de precisão com pequenos valores de distância, não comprometendo desta forma o funcionamento do sistema, já que as coordenadas usadas são limitadas em quatro casas decimais. Também ocorrem, algumas vezes, problemas de captação de sinal pelo GPS (quando se trata de lugares fechados).

Para trabalhos futuros sugere-se a implementação de busca binária dentre as coordenadas do arquivo, possibilitando que as coordenadas sejam verificadas mais rapidamente. Também é desejado que seja implementada uma forma de efetuar a liberação e o bloqueio da trava através e uma página *web*, possibilitando executar tais funcionalidades sem que seja necessário encontrar-se em um dos pontos de entrega.

## Referências


Arduino (2015). https://www.arduino.cc/en/Main/ArduinoBoardMega2560#, [accessed on Sep 12].

Arnizaut, G. (2014). Sistema de Monitoramento e Registro de Dados para Apoio à Navegação de um Veículo Solar. Universidade do Rio de Janeiro.

Banzi, M. (2009). *Getting Started Wiht Arduino*. 1. ed. O'Reilly Media.

Bentes, L. M. A. (2013). Sistema de Segurança Veicular Com Uso de GPS Baseado em Arduino. Universidade do Estado do Amazonas - UEA.

Booch, G., Rumbaugh, J. and Jacobson, I. (2000). *UML: Guia do Usuário*. 1. ed.

Cavalcante, M. A., Tavolaro, C. R. C. and Molisani, E. (2011). Física com Arduino para Iniciantes. *Revista Brasileira de Ensino de Física*, v. 33, p. 9.

Correios (2015). http://www.correios.com.br/sobre-correios/a-empresa/quem-somos/principais-numeros, [accessed on Sep 12].

D'Alge, J. C. L. (1991). Cartografia para Geoprocessamento. *Introdução à Ciência da Geoinformação*. http://www.dpi.inpe.br/gilberto/livro/introd/index.html.

Da bússola ao GPS (2011). http://redeglobo.globo.com/globociencia/noticia/2011/10/da-bussola-ao-gps-entenda-como-funciona-navegacao-na-era-digital.html, [accessed on Oct 12].

Davis, C. and Neto, G. C. (2001). Arquitetura de sitemas de informacao geografica. Instituto Nacional de Pesquisas Espaciais.

Druk, S., Carvalho, M. S., Câmara, G. and Monteiro, A. M. V. (2004). *Análise Espacial de Dados Geográficos*. 1. ed. Embrapa.

Ferreira, J. P. L. (2005). Open Source Software. Universidade de Coinbra, Portugal.

Gibson, R. and Schuyler, E. (2006). *Google Maps Hacks*. 1. ed. O'Reilly Media.

Gioppo, L. L., Higaskino, M. M. K., Costa, R. F. and Meira, W. H. T. (2009). Robô Seguidor de Linha. Universidade Tecnológica Federal do Paraná.

Lima, M. S., Casillo, L. and Casillo, D. (2014). Desenvolvimento de uma Célula Braille de Baixo Custo Usando Arduino. Congresso Brasileiro de Automática.

Monico, J. F. G. (2001). *Posicionamento Pelo NAVSTAR-GPS*. 1. ed. Unesp.

Palmer, S. R. and Felsing, J. M. (2002). *A Practical Guide to Feature-Driven Development*. 1. ed. Prentice Hall.

Pappis, R. M. and Prass, F. S. (2012). Sistema GPS Android. ULBRA-SM.

Purvis, M., Sambells, J. and Turner, C. (2006). *Google Maps Applications with PHP and Ajax: From Novice to Professional*. 1. ed. Apress.

Retamal, A. M. (2014). FDD - Feature Driven Development. http://www.heptagon.com.br/fdd, [accessed on Oct 10].

Santos, J. and Pena, H. (2011). Geoprocessamento Aplicado a Ecologia de Paisagem: Uma Análise Dinâmica Espacial da Ilha do Papagaio - PA, Amazônia - Brasil. Universidad de Málaga.

Sram, I. and Cycles, W. E. (2014). Atmel ATmega640/V-1280/V-1281/V-2560/V2561/V.