UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO – BACHARELADO

PROTÓTIPO DE UM SISTEMA DE RASTREAMENTO VEICULAR BASEADO NO MÓDULO TELIT

LEANDRO BESZCZYNSKI

BLUMENAU
2008

2008/1-23

LEANDRO BESZCZYNSKI

PROTÓTIPO DE UM SISTEMA DE RASTREAMENTO VEICULAR BASEADO NO MÓDULO TELIT

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciências da Computação — Bacharelado.

Prof. Miguel Alexandre Wisintainer, Orientador

BLUMENAU
2008

2008/1-23

PROTÓTIPO DE UM SISTEMA DE RASTREAMENTO VEICULAR BASEADO NO MÓDULO TELIT

Por

LEANDRO BESZCZYNSKI

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

	_____
Presidente:	Prof. Miguel Alexandre Wisintainer, Mestre e Orientador- FURB
	_____
Membro:	Prof. Antonio Carlos Tavares, Mestre - FURB
	_____
Membro:	Prof. Francisco Adell Péricas, Mestre - FURB

Blumenau, 09 de julho de 2008

AGRADECIMENTOS

A Deus por me dar força nos momentos de dificuldades.

Ao professor e orientador Miguel Alexandre Wisintainer, por acreditar na realização deste, além de toda atenção e apoio dispensados.

À minha namorada Beatriz, que com muito incentivo e motivação sempre esteve presente.

À minha família, que nunca deixou de prestar o suporte necessário para que eu pudesse chegar à condição de formando.

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

RESUMO

Este trabalho apresenta o estudo e desenvolvimento de um sistema de rastreamento veicular, que tem como objetivo principal a coleta e transmissão das informações de localização do veículo através do módulo Telit para um computador servidor, permitindo assim que as informações sejam disponibilizadas para posterior consulta via web, proporcionando segurança e mobilidade aos proprietários de veículos.

Palavras-chave: Rastreamento. Segurança. Mobilidade.

ABSTRACT

This work presents the study and development of tracking system running, which has as its main objective the collection and transmission of information to locate the vehicle through Telit module to computer server, thus allowing the information is made available for further consultation via web, providing security and mobility to owners of vehicles.

Key-words: Tracking. Security. Mobility.

LISTA DE ILUSTRAÇÕES

LISTA DE SIGLAS

EEPROM – *Electrically Erasable Programmable Read Only Memory*

FTP - *File Transfer Protocol*

GPRS - *General Packet Radio Service*

GPS - *Global Positioning System*

GSM - *Global System for Mobile communications*

I2C ou IIC - *Inter-Intergrated Circuit*

I/O - *Input/Output*

LCD - *Liquid Crystal Display*

PC - *Personal Computer*

PDAs - *Personal Digital Assistants*

RAM - *Random Access Memory*

RF - Requisito Funcional

RNF - Requisito Não Funcional

SIM - *Subscriber Identity Module*

SMS - *Short Message Service*

SPI - *Serial Peripheral Interface*

TCP/IP - *Transmition Control Protocol/Internet Protocol*

USB - *Universal Serial Bus*

3G - Terceira Geração para telefonia móvel

SUMÁRIO

1 INTRODUÇÃO

> A segurança falha oferecida pelas instituições oficiais, que tem se mostrado incompetente para evitar o crescimento do número de furtos e roubos de cargas e veículos, tem levado pessoas e empresas a procurar por conta própria uma forma de proteger seus patrimônios. (BELÓRIO, 2005, p. 10).

Diante dessa afirmação de Belório, percebe-se que as empresas estão aperfeiçoando-se tecnologicamente, para desta forma garantir maior segurança e assim obter diferencial diante da globalização do mercado. Exemplo disso seria uma empresa que presta serviço de transporte de cargas, onde se faz indispensável o fornecimento de precisão, rapidez, segurança e possibilidade de localização imediata do veículo durante todo o processo de prestação de serviço.

Belório (2005, p. 10) acrescenta que o aumento da violência e furto de veículos, nos grandes centros urbanos, já atinge grande parcela da população e atualmente já é uma preocupação até mesmo de montadoras de veículos, que estudam possibilidades de implantação de um sistema de rastreamento em veículos novos.

De acordo com Câmara (2000, p. 1), prevendo o futuro, leva a imaginar a possibilidade da existência de um mundo no qual todos os componentes de relevância (coisas ou pessoas) possuam uma identidade e até possam tornar, a qualquer momento, sua localização possível, algo como um aparelho compacto de *Global Positioning System* (GPS) com um código de identificação pessoal. Câmara (2000, p. 2) ainda acrescenta que existe uma crescente e contínua expansão na utilização de dispositivos de conexão sem-fio (*wireless*) no mercado de produtos e serviços, significando assim cada vez mais a necessidade desses para disponibilizar localização imediata.

Com base na necessidade de obter localização imediata e proporcionar tranqüilidade tanto para pessoas físicas quanto jurídicas, iniciou-se o estudo em busca do desenvolvimento de uma solução de rastreamento geográfico. O principal objetivo é oferecer uma nova possibilidade de localização para diversos fins, como ferramenta para auxiliar na tomada de decisão de empresas de transportes ou para localizar um automóvel ou carga que fora roubado e até mesmo para auxiliar autoridades na localização de rotas de quadrilhas de roubo de veículos, bem como na identificação do motorista.

Diante das necessidades apresentadas, iniciou-se a busca por um componente que pudesse suprir boa parte destas, assim, optou-se por um módulo de localização e comunicação chamado Telit. Esse módulo proporciona maior facilidade no desenvolvimento do software

embarcado no protótipo, pois possui interpretador de comandos Python e esta é uma linguagem difundida mundialmente além de possuir vasta biblioteca de funcionalidades de programação e comunicação. O módulo Telit será devidamente descrito em capítulo específico.

1.1 OBJETIVOS DO TRABALHO

O objetivo deste trabalho é desenvolver uma solução para rastreamento de veículos proporcionando segurança e mobilidade a seus proprietários.

Os objetivos específicos são:

a) registrar as informações referentes à localização do veículo em qualquer parte do mundo que possua cobertura GSM;
b) armazenar uma imagem (foto) do interior do veículo capturada pelo protótipo proposto no instante do fornecimento da localização;
c) disponibilizar, através de uma página web, consultas das localizações e fotos registradas pelo protótipo proposto;
d) efetuar bloqueio do veículo através de comando remoto.

1.2 ESTRUTURA DO TRABALHO

Este trabalho divide-se em quatro capítulos, estando distribuídos da forma que se segue:

No primeiro capítulo, descreve-se a introdução ao assunto e os objetivos desejados. No segundo capítulo apresenta-se a fundamentação teórica que se faz imprescindível para o desenvolvimento deste. O terceiro capítulo trata das especificações necessárias, implementações desenvolvidas, ferramentas utilizadas e rotinas de testes realizados. Finalmente o quarto capítulo descreve as considerações finais e conclusões obtidas, deixando sugestões para o desenvolvimento de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os conceitos relevantes para o desenvolvimento deste trabalho, os quais são: sistema GPS, rede GSM, linguagem de programação Python, ferramenta PythonWin, módulo Telit, ferramenta para atualização de *scripts* no módulo Telit, protocolo TCP/IP, protocolo I2C, dispositivo escravo I2C, software embarcado e trabalhos correlatos.

2.1 SISTEMA GPS

> O GPS (Global Positioning System ou Sistema de Posicionamento Global) é um sistema de posicionamento espacial baseado em rádio navegação [...] tem como princípio a medida da distância entre a antena do satélite e a do receptor e isso independe, dentro de certos limites, das condições meteorológicas. (DRAGO; DISPERATI, 1996, p. 1).

Conforme Drago e Disperati (1996, p. 1), o sistema GPS teve seu desenvolvimento iniciado a partir de 1973, nos Estados Unidos, onde o uso seria de exclusividade militar.

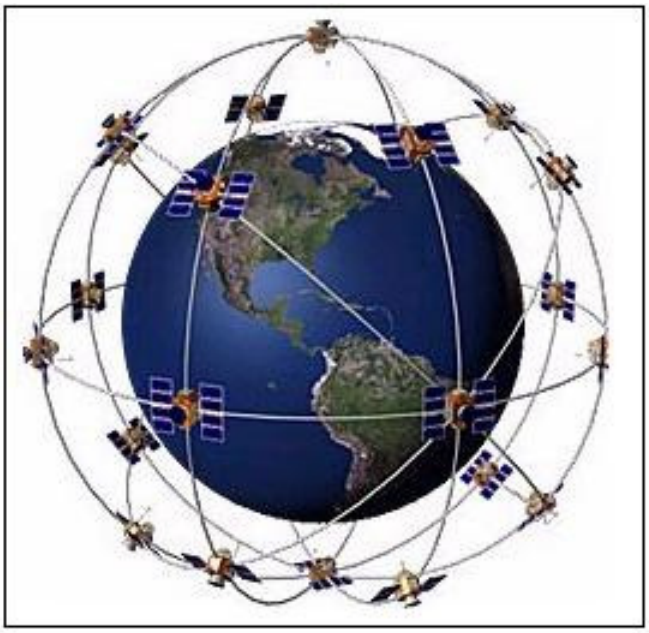
De acordo com Rocha (2003, p. 11), os sistemas de posicionamento global surgiram para fornecer posicionamento em qualquer lugar do mundo. Este sistema representa uma revolução quando se fala em técnicas de navegação e mensuração. Anteriormente eram utilizados, para esse fim, bússolas e teodolitos. Com esses aparelhos era possível se obter direções de forma direta e distâncias indiretamente, com as quais se conseguia calcular coordenadas e áreas.

Ainda segundo Rocha (2003, p. 11), os sistemas de posicionamento resolvem, de forma instantânea, problemas de navegação (“onde estou”, “para onde vou” e “tempo de percurso”) e também de mensuração (cálculo de coordenadas, azimutes/rumos, distâncias e áreas), fornecendo para isso direção, posição, velocidade, área e distância.

Belório (2005, p. 12), salienta que o sistema GPS é composto por uma rede de 24 satélites colocada em órbita pelo Departamento Norte Americano de Defesa, conforme pode-se visualizar na Figura 1. O GPS trabalha em qualquer condição de tempo, em qualquer lugar do mundo, 24 horas por dia, e não há cobrança de nenhuma taxa para utilizá-lo.

Belório ainda acrescenta que um receptor de GPS deve receber sinais de pelo menos

três dos 24 satélites para calcular uma posição 2D (latitude e longitude) e movimento de rastro. Com quatro ou mais satélites visíveis, o receptor poderá determinar a posição 3D do usuário (latitude, longitude e altitude). Certos fatores atmosféricos e outras fontes de erro podem afetar a precisão dos receptores, normalmente no máximo 15 metros de imprecisão.



Fonte: Belório (2005, p. 13).

Figura 1 – Configuração do sistema GPS

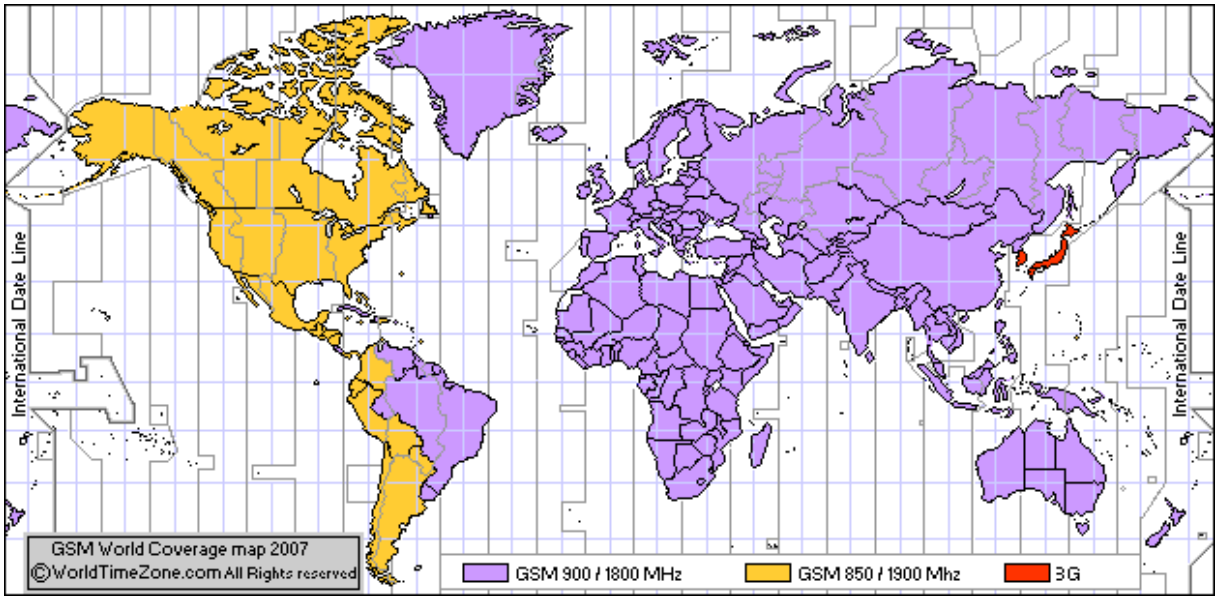
2.2 REDE GSM

Segundo Sverzut (2005, p. 59), antes mesmo da década de 80, a Europa era tomada por um mercado de celulares caracterizado por um grande número de padrões analógicos incompatíveis e limitados. Baseando-se nisso, as autoridades européias, viram a necessidade da implantação de um sistema digital de telefonia celular, surgia assim a tecnologia GSM.

Sverzut (2005, p. 59) afirma que o GSM é uma excelente tecnologia, com grande e rápida expansão. Por volta do ano 2000, havia algumas poucas dezenas de empresas que trabalhavam com GSM. Por volta de 2005 esse número já ultrapassava centenas de empresas e milhares de especialistas que trabalhavam com essa tecnologia.

Uma evolução das redes GSM, segundo Silva e Moreira (2005, p. 50), é o GPRS, um serviço de comunicação que permite uma conexão a Internet sem a necessidade de estabelecer uma chamada telefônica para transferência de dados, incluindo dessa forma suporte a serviços de dados com os protocolos TCP/IP.

Na Figura 2 pode-se visualizar a cobertura GSM em todo o mundo, destacando que a nova tecnologia 3G (Terceira geração para telefonia sem fio) já é realidade no Japão e Coréia do Sul.



Fonte: adaptado de WorlTimeZone (2007).

Figura 2 – Mapa mundial e cobertura GSM

2.3 LINGUAGEM DE PROGRAMAÇÃO PYTHON

> Python é uma linguagem fácil e poderosa. Possui mecanismos eficientes e com um bom nível de abstração para manipulação de estruturas de dados. É uma linguagem interativa, interpretada e orientada a objetos com uma sintaxe elegante e tipagem dinâmica [...] algumas pessoas falam que Python é uma linguagem mágica porque você consegue fazer quase tudo com um mínimo esforço. (CATUNDA, 2001, p. 6).

De acordo com Catunda (2001, p. 6), a linguagem Python é muito utilizada e difundida, própria para desenvolvimento de *script*[1] e também muito conhecida por ser rápida quando destinada à soluções web e até programas gráficos de diversas áreas e plataformas.

Segundo Lutz (2001, p. 1), Python enfatiza conceitos como a qualidade, produtividade, portabilidade e integração. Destaca principalmente que Python permite escrever softwares de forma extremamente fácil, rápida e de manutenção favorecida pela legibilidade proporcionada.

[1] Considerados como programas que consistem em uma seqüência de instruções de uma linguagem para emprego em determinado aplicativo.

2.4 FERRAMENTA PYTHONWIN

Hammond (1999) cita PythonWin como ferramenta auxiliar para desenvolvimento e depuração de *scripts* Python. Este software possui um rico editor para códigos Python, detectando e indicando visualmente as palavras reservadas da linguagem.

Outra facilidade detectada neste pacote Python de Hammond, é a facilidade no desenvolvimento, visto que a ferramenta realiza detecção de erros através de um validador de *scripts*, indicando visualmente o erro encontrado. É importante também destacar a possibilidade de depuração de *scripts*.

Na Figura 3, pode-se visualizar a interface desta ferramenta.

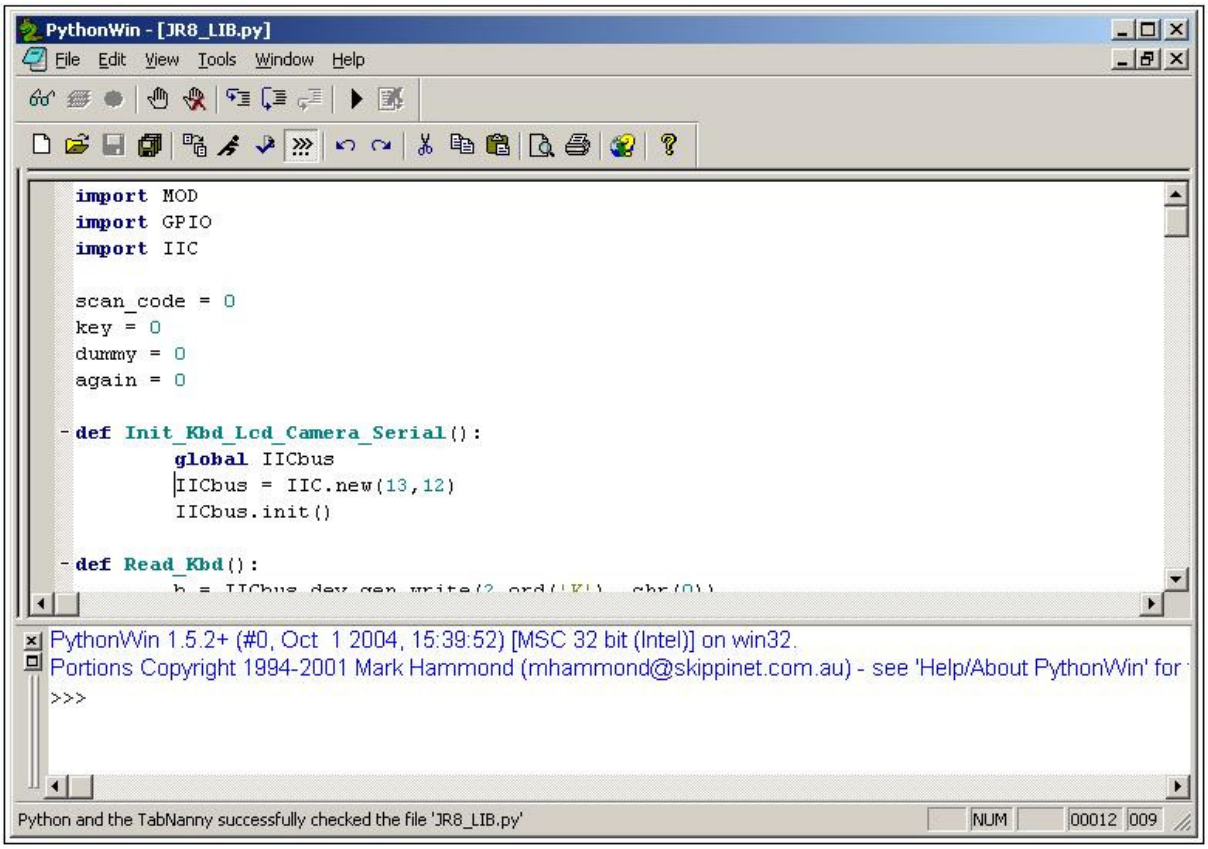


Figura 3 - Editor e interpretador de *scripts* Python

2.5 MÓDULO TELIT

Segundo Telit (2007), o módulo GM862-GPS, possui funcionalidades que podem ser

aplicáveis a sistemas automotivos de segurança, gerência de frotas, telemetria e controle e sistemas de segurança em geral. Ainda de acordo com Telit (2007), o módulo possui localizador GPS, capacidade de comunicação via GSM, compatibilidade com protocolo TCP/IP, GPRS e possui interpretador de *scripts* Python.

Telit (2007) ainda acrescenta como vantagem a pequena dimensão do componente, sendo de 43,9mm X 43,9mm, espessura de 6,9mm e peso de 19g, estendendo ainda mais as possibilidades de aplicações do mesmo.

Como característica importante, Telit (2007) cita o interpretador interno de *scripts* Python, que permite a gravação dos códigos desenvolvidos diretamente na linguagem utilizada pelo usuário. Para integrar este interpretador, o módulo já possui internamente uma vasta variedade de bibliotecas dedicadas a funções do módulo, concentrando dessa forma, funcionalidades específicas para cada objetivo, permitindo grande legibilidade dos *scritps* desenvolvidos. Abaixo segue bibliotecas e suas respectivas funcionalidades:

a) MDM – Utilizada para o envio e recebimento dos comandos AT para o modem interno;
b) SER – Destinada ao envio e recebimento de comandos pela porta serial, podendo ser utilizada para leitura de dispositivos externos ou para rastreamento e eliminação de erros;
c) GPIO – É responsável pelo gerenciamento direto dos pinos de entrada e saída para uso geral;
d) MOD – É composta de um conjunto de geral de funções úteis. Por exemplo: *sleep*(tempo pausa);
e) IIC – É utilizada para criar e gerenciar portas de comunicação com o protocolo I2C criadas a partir de pinos de GPIO;
f) SPI – Permite criar e gerenciar portas de comunicação com o protocolo SPI criadas a partir de pinos de GPIO;
g) GPS – Dedicada a utilização das funções do controlador de GPS no módulo.

Na Figura 4 pode-se visualizar o módulo Telit.

Fonte: Telit (2007).

Figura 4 – Módulo Telit GM862-GPS

2.6 FERRAMENTA PARA ATUALIZAÇÃO DE *SCRIPTS* NO MÓDULO TELIT

Santis (2006) destaca que o uso da ferramenta SxPyDownloadTool é destinado a facilitar a transferência dos *scripts* escritos em linguagem Python para os produtos Telit que possuem interpretador Python.

Santis ainda destaca que a ferramenta permite realizar *upload* de novos *scripts* e exclusão de *scripts* já gravados, além de permitir a ativação do *script* principal que será executado pelo modem.

Santis (2006) reforça que o software necessita de um microcomputador padrão equipado com Windows 2000 ou XP e de porta serial ou conversor USB para comunicação. Depois de conectar fisicamente e iniciar o software, pode-se iniciar a comunicação com o botão conectar. Depois disso, os *scripts* e arquivos de memória são listados na caixa à direita, permitindo realizar as operações citadas anteriormente.

Na Figura 5 visualiza-se o software conectado ao módulo Telit.

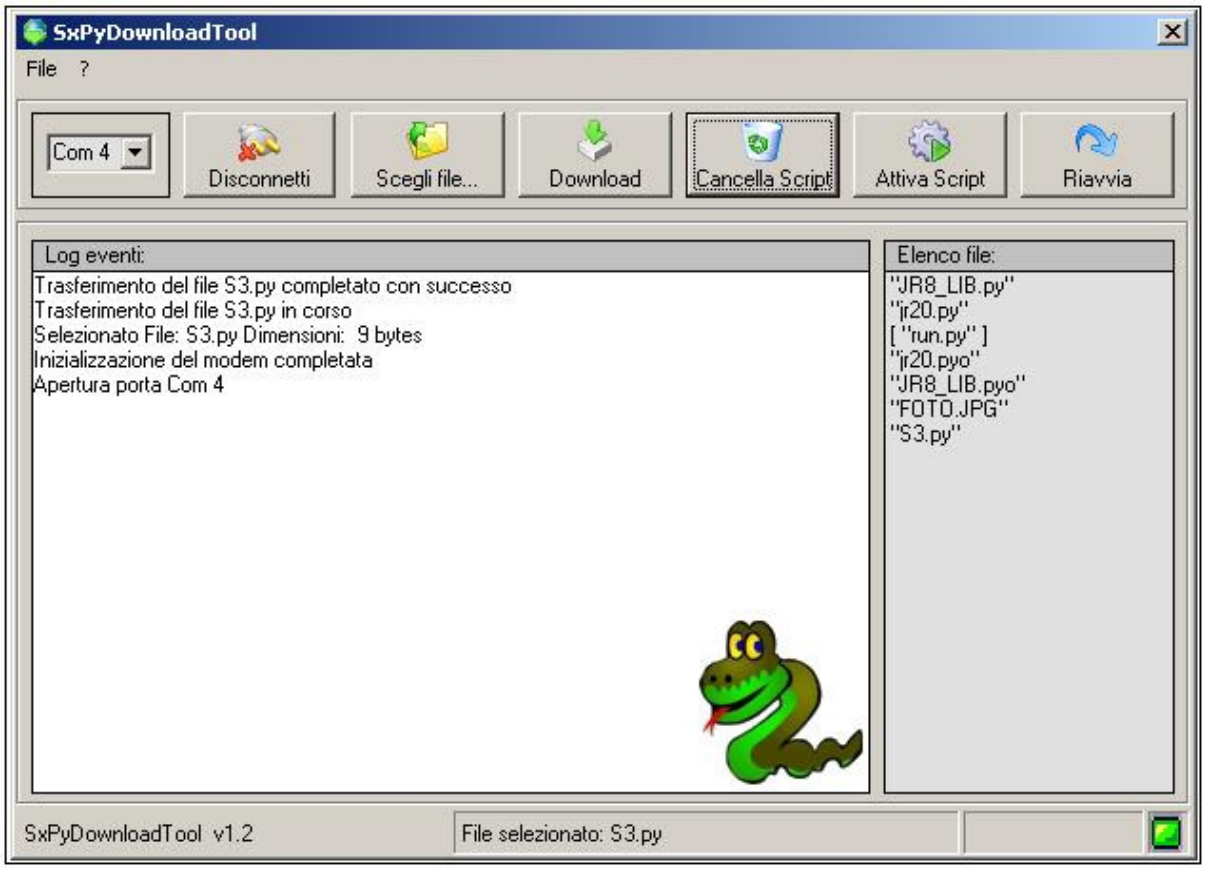


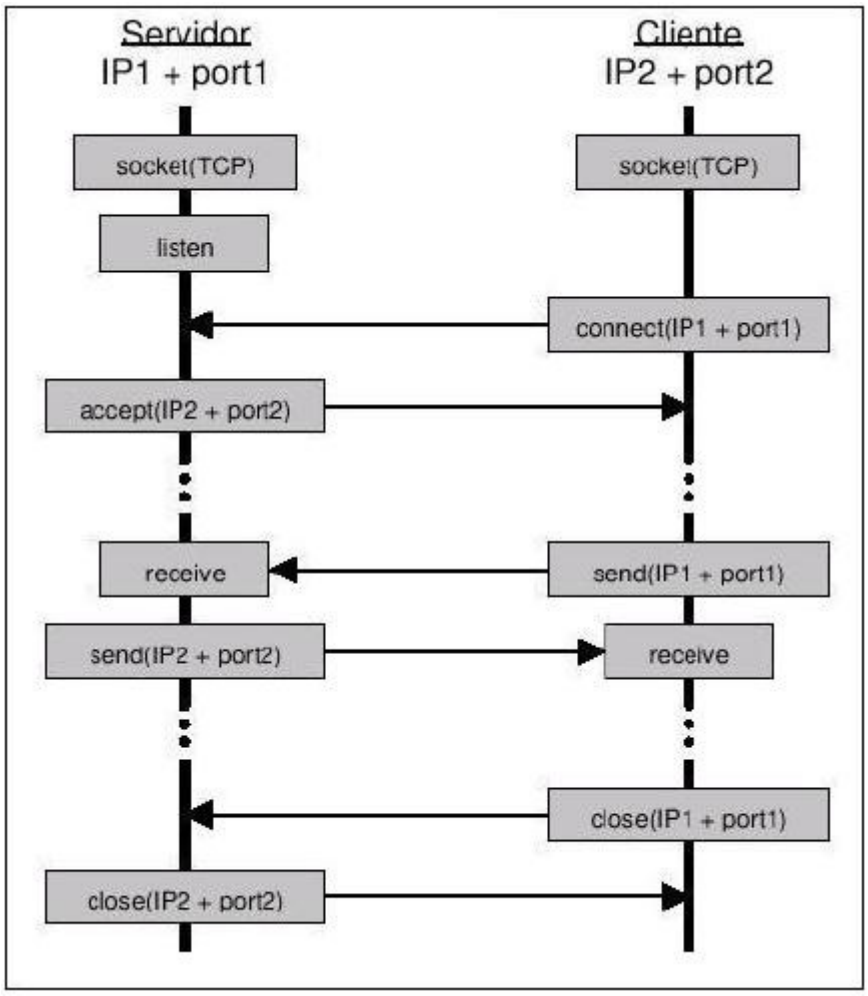
Figura 5 - Ferramenta de atualização e ativação de *scripts* no módulo Telit

Esta ferramenta proporciona rápido resultado dos *scripts* desenvolvidos, visto que logo depois de salvar o código no microcomputador, utiliza-se esta ferramenta para fazer a gravação do código no módulo Telit, e assim pode-se visualizar o resultado prático já com o módulo telit interpretando o *script* gravado.

2.7 PROTOCOLO TCP/IP

Conforme afirma Montibeller (2005, p. 17), “Transmission Control Protocol (TCP) é um protocolo orientado a conexão que fornece um serviço confiável de troca de dados fim-a-fim.”. O TCP/IP é um protocolo de controle para transmissão de dados muito confiável e isso o tornou padrão mundial para conexão de redes.

Segundo Seixas Filho (2007, p. 2), “TCP/IP é na verdade o nome genérico para uma família de protocolos e utilidades também conhecido por *Internet Protocol Suite*, onde *Suite* designa uma pilha (*stack*) de protocolos.”. Na Figura 6 pode-se visualizar o processo de troca de mensagens entre cliente e servidor.



Fonte: Péricas (2003, p. 93).

Figura 6 – Estabelecimento de um *socket* para troca de mensagens TCP

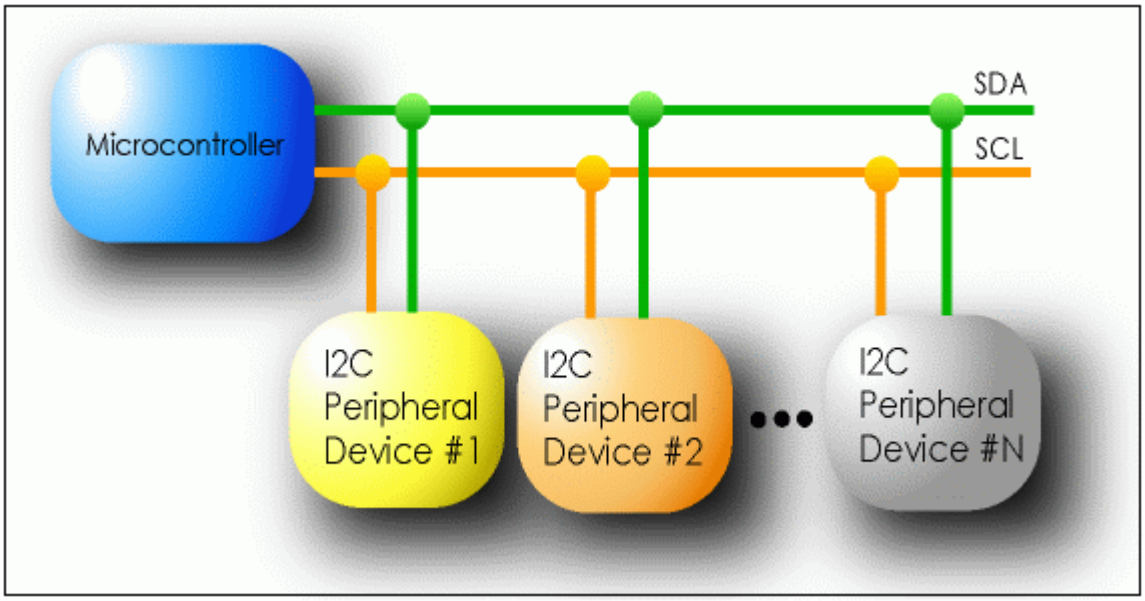
Péricas (2003, p. 92) associa o TCP/IP como um canal virtual de comunicação entre um *socket*[2] cliente e um *socket* servidor, e para garantir a entrega confiável das informações, o principal componente desse protocolo é a camada de transporte.

2.8 PROTOCOLO I2C

De acordo com Silva e Pinho (2007, p. 30), este protocolo de comunicação, I2C ou IIC, foi desenvolvido pela Philips em 1996 e hoje encontra-se amplamente difundido, permitindo interligação de ampla gama de dispositivos eletrônicos. Destes dispositivos, Silva e Pinho citam microcontroladores, microprocessadores e circuitos de uso geral, portas de I/O, memórias RAM e EEPROM ou conversores de dados.

[2] Considerado como repositório onde pode-se enviar mensagens e a partir do qual podem ser recebidas.

Na Figura 7 pode-se visualizar a disposição de periféricos em um barramento de comunicação I2C.



Fonte: adaptado de UCPROS (2008).

Figura 7 – Barramento de comunicação I2C e periféricos

2.9 DISPOSITIVO ESCRAVO I2C

De acordo com Basic4Ever (2008), o dispositivo tem como principal objetivo prover comunicação facilitada para componentes adicionais a qualquer projeto eletrônico. O dispositivo escravo I2C permite gerenciamento autônomo de teclado 4x4, sensor de pânico, câmera serial e LCD de até 4 linhas por 20 colunas.

Basic4Ever (2008) destaca que todo o controle do dispositivo escravo é realizado através de simples comandos I2C, endereçando cada função com uma identificação única no dispositivo e este gerencia toda a comunicação dos periféricos conectados a ele.

2.10 SOFTWARE EMBARCADO

Taurion (2005, p. 1) afirma que o software embarcado é qualquer software que se encontra embutido em um equipamento de qualquer natureza. Taurion ainda cita como

exemplo o sistema de injeção eletrônica automotiva, onde o software embarcado é responsável pelo gerenciamento e flexibilidade do sistema. Outros exemplos também são citados pelo mesmo autor, como *Personal Digital Assistantes* (PDAs), microondas e televisores.

Segundo Kondo et al. (2006, p. 1), o software embarcado pode ser também conhecido como software embutido, do inglês "*Embedded Software*", pois é um sistema computacional embutido em um sistema maior, programado especificamente para uma tarefa.

Kondo ainda salienta que os sistemas embarcados representam as melhores oportunidades de economia emergente no Brasil, visto que possuem o intuito de proporcionar maior interoperabilidade em ambientes heterogêneos de software, hardware e plataformas distintas dos mais diversificados dispositivos.

2.11 TRABALHOS CORRELATOS

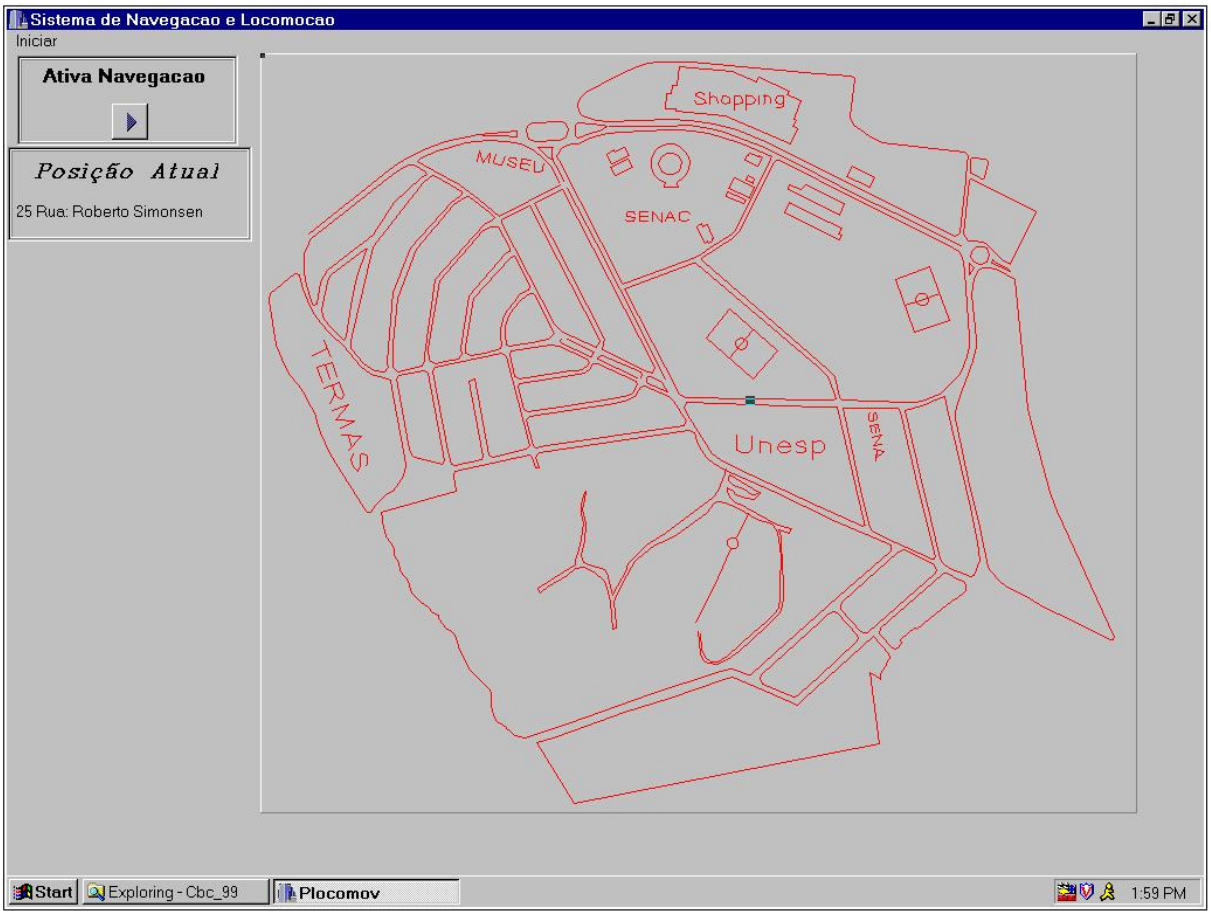
Existem soluções que propõem-se a desempenhar papéis semelhantes ao proposto no presente trabalho. Dentre estas, foram selecionadas: Sistema de Localização e Navegação Apoiado por GPS (HASEGAWA et al., 1999), Implementação do Sistema de Mapeamento de uma Linha de Ônibus para um Sistema de Transporte Inteligente (WEIGANG et al., 2001) e Descrição de um Sistema de Rastreamento Veicular Utilizando GPS (BELÓRIO, 2005).

2.11.1 Sistema de localização e navegação apoiado por GPS

Hasegawa et al. (1999, p. 1) dedicou especial interesse ao estudo de navegação, citando inclusive cidades dos Estados Unidos, Japão e outros países da Europa que os muitos carros de passeio contam com um sistema de localização, com o objetivo de auxiliar o motorista durante seu trajeto. Para que isso se transformasse em realidade também aqui no Brasil, Hasegawa (1999) desenvolveu um protótipo de um sistema de navegação, utilizando dados provenientes de um receptor GPS e integrando essas informações em uma base de dados geográficos em um computador portátil.

As informações podem ser visualizadas na tela do computador, mostrando assim a posição imediata do veículo, sobre um mapa geográfico que caracteriza a região,

possibilitando assim a identificação até mesmo da rua atual, que pode ser expandido inserindo maiores informações e detalhes de cada região específica, como pode ser visto na Figura 8.



Fonte: Hasegawa (1999).

Figura 8 – Tela do protótipo de navegação

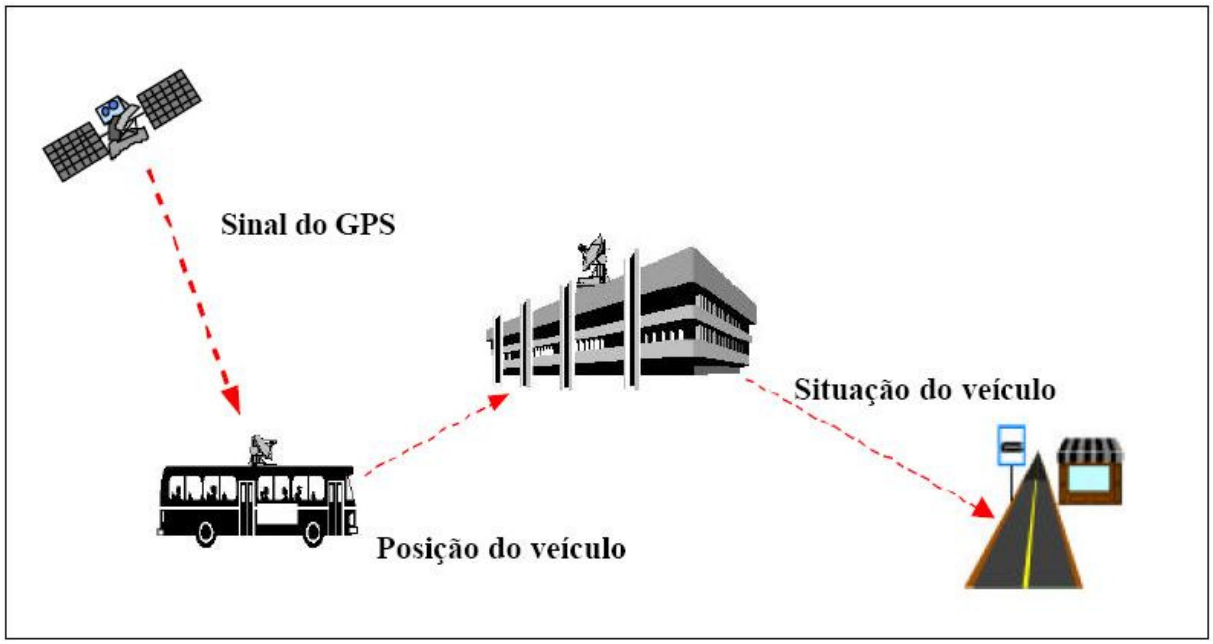
2.11.2 Implementação do sistema de mapeamento de uma linha de ônibus para um sistema de transporte inteligente

De acordo com Weigang et al. (2001, p. 1), o transporte coletivo atual tem fundamental importância no dia a dia da população brasileira, representa em muitos casos, o único meio de transporte familiar. Weigang destaca que devido ao descontentamento da população com esta forma de transporte, os congestionamentos de veículos nas vias públicas tornaram-se inevitáveis.

Weigang et al. (2001, p. 2) sugere que para auxiliar na programação e satisfação dos usuários de transporte coletivo público, desenvolva-se um sistema de informação sobre os horários individuais por ônibus.

Para que isso ocorra, Weigang destaca a necessidade da utilização de tecnologias de

telefonia celular, satélites do sistema GPS, internet. Pode-se visualizar a comunicação entre os equipamentos propostos através da Figura 9.

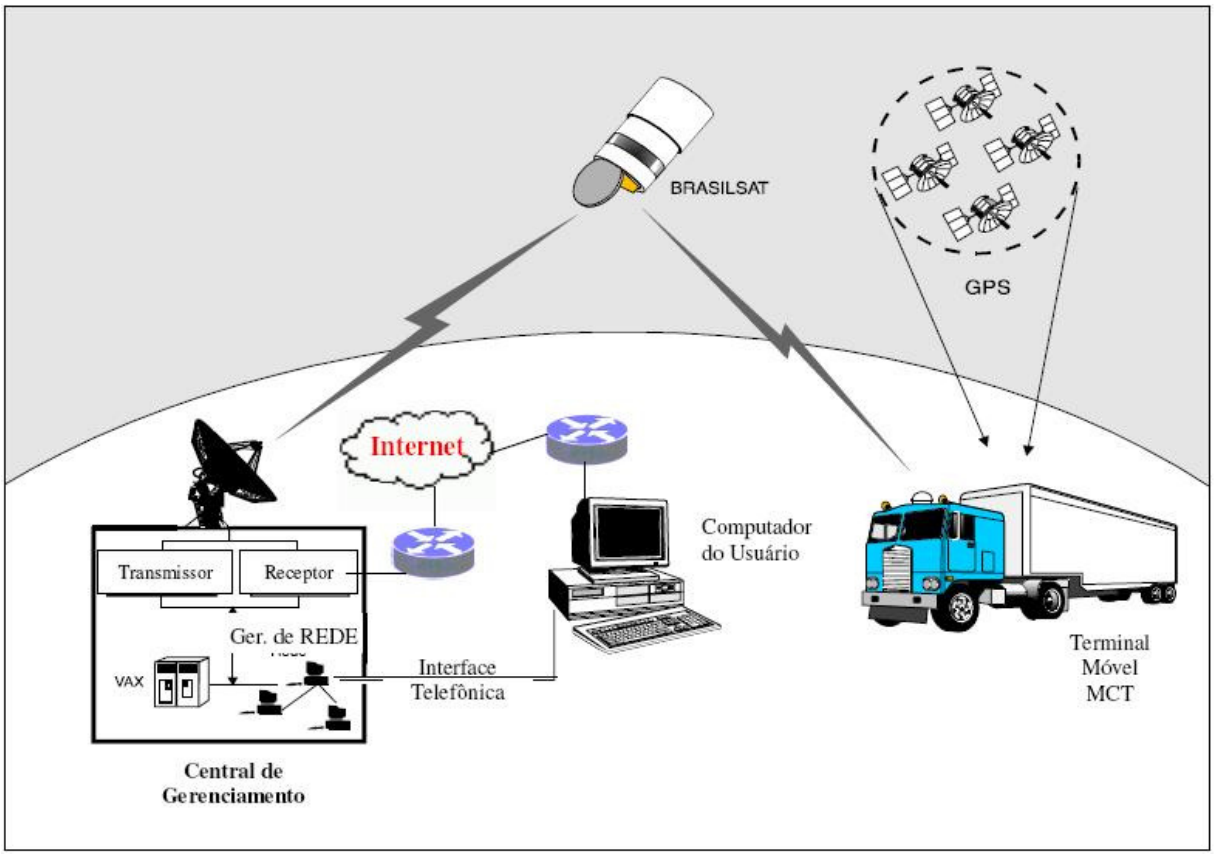


Fonte: Weigang et al. (2001).

Figura 9 – Diagrama do protótipo

2.11.3 Descrição de um sistema de rastreamento veicular utilizando GPS

Segundo Belório (2005, p. 10) em sua descrição sobre o funcionamento de um sistema rastreador veicular que utiliza GPS, atualmente, para a maioria dos sistemas de rastreamento, é instalado um comunicador no interior do veículo que possui um receptor GPS, fornecendo assim as informações de latitude, longitude, altitude, data e hora. Estas informações são originadas a partir do recebimento de sinais enviados por satélites. O receptor realiza cálculos sobre as informações para enfim obter os dados de localização, em seguida repassa as informações para um centro de controle, que envia um e-mail diretamente ao cliente, conforme diagrama ilustrado na Figura 10.



Fonte: Belório (2005, p. 21).

Figura 10 – Diagrama dos componentes do sistema

Ainda segundo Belório (2005, p. 23), o sistema de rastreamento pode ser extendido com diversos controles adicionais, como sensores, travas e imobilizadores, todos esses, gerenciados por um computador de bordo. Esses controles adicionais, chamados atuadores, são muito importantes para empresas que necessitam de ferramentas de gerenciamento de risco, principalmente para transportes de cargas.

2.11.4 Considerações sobre os trabalhos correlatos

No quadro abaixo são apresentadas as principais características sobre os trabalhos de Hasegawa (1999), Weigang (2001) e Belório (2005).

Características	**Hasegawa (1999)**	**Weigang (2001)**	**Belório (2005)**
Implementação em Software	*	*	
Implementação em Hardware			
Captura de imagem			
Visualização via web		*	*
Tempo Real	*	*	*
Comunicação	Serial	GSM	Satélite Banda C
Linguagem de Programação	C	JAVA	Não Informado

Quadro 1 – Características dos trabalhos correlatos

3 DESENVOLVIMENTO

Neste capítulo são detalhados os requisitos do sistema, a especificação de hardware e software, implementações realizadas e os resultados obtidos.

3.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

A ferramenta deverá conter os seguintes Requisitos Funcionais (RF) e Requisitos Não-Funcionais (RNF):

a) o sistema deverá registrar informações de localização global do veículo em uma base de dados para posterior consulta (Requisito Funcional - RF);
b) o sistema deverá registrar uma imagem do motorista, capturada no interior do veículo no momento que a localização foi obtida pelo protótipo (RF);
c) o sistema deverá atuar sobre circuitos externos de segurança ou bloqueio do veículo, através de comandos remotos ou previamente programados no software embarcado (RF);
d) o sistema deverá ser implementado utilizando linguagem Python (Requisito Não-Funcional - RNF);
e) o sistema deverá utilizar banco de dados SQLServer para armazenar as informações (RNF);
f) a página web deverá consultar o banco de dados e disponibilizar as informações de localização do veículo e imagens registradas para consulta (RNF);
g) a página web deverá ser implementada em PHP (RNF);
h) o sistema deverá estar disponível para utilização a qualquer instante (RNF);
i) para utilização do sistema, deverá ser disponibilizado um chip celular GSM para autenticação e envio das informações através do serviço GPRS (RNF).

3.2 ESPECIFICAÇÃO

O protótipo desenvolvido divide-se em duas partes distintas. A primeira a ser especificada será o hardware, que é o protótipo a ser instalado no veículo. A segunda parte será o software, que ainda pode ser subdividido em três softwares distintos. Um será o software embarcado no módulo Telit, o outro será o software instalado em um microcomputador que receberá e registrará as informações enviadas pelo protótipo e ainda a página web que fará as consultas para o usuário através da internet.

3.2.1 Hardware

O hardware tem como funções principais o fornecimento da informação de posicionamento global, detecção de pânico, a captura de uma imagem do motorista, o controle de corte de alimentação para realizar o bloqueio do veículo, além de realizar a conexão GPRS para o envio das informações obtidas para o software que estará disponível para registrar as mesmas em um computador servidor.

Este hardware é constituído de 2 partes fundamentais: módulo Telit e dispositivo escravo I2C. O dispositivo escravo I2C fará o gerenciamento de um *Liquid Crystal Display* (LCD), uma câmera para captura de imagens e o sensor de pânico para situações não previstas. Pode-se visualizar o esquemático na Figura 11.

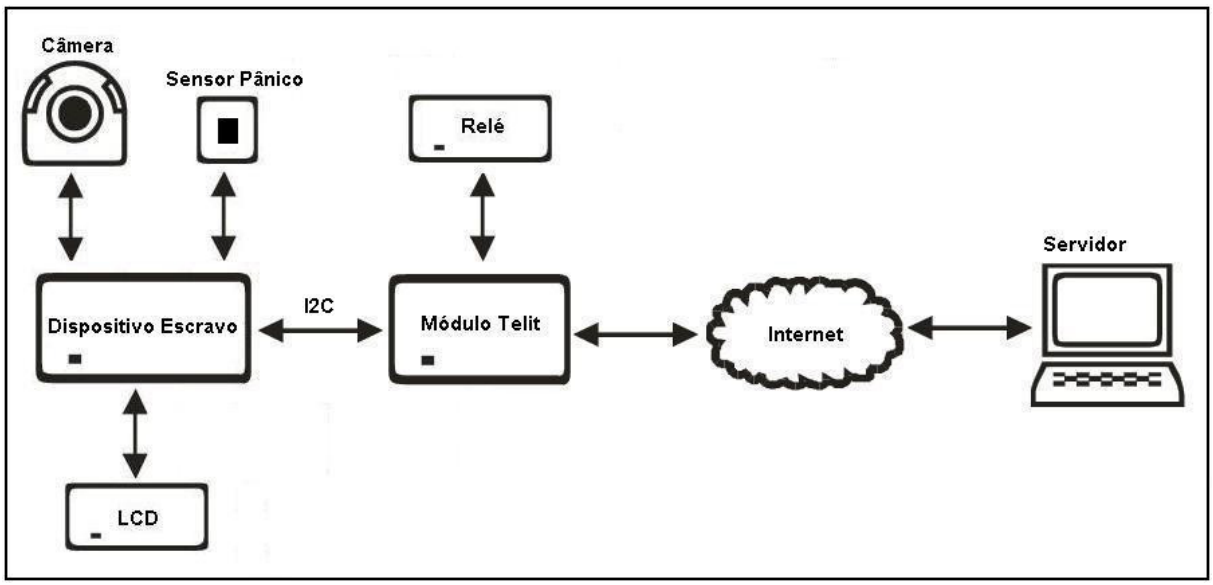


Figura 11 – Ligação entre os componentes do sistema

Para a construção do protótipo proposto, foi utilizada uma placa fornecida pela empresa Seva (2008), específica para utilização e acoplamento do componente GM862 da Telit (2007), tanto que este último é fornecido também pela empresa Seva, que é representante da Telit no Brasil.

Adicionou-se também ao protótipo uma placa padrão de medidas 10cm x 20cm para confecção de circuitos diversos, esta, foi utilizada para ligações adicionais entre o módulo Telit e dispositivo escravo I2C.

As informações adquiridas pelo protótipo, deverão ser registradas em uma base de dados que ficará disponível para consultas através de uma página web especificada em uma próxima seção. O protótipo fará uma conexão GPRS com o servidor utilizando o modem interno, registrando as informações de localização através de um software instalado no computador servidor por uma porta disponibilizada para este fim. O protótipo também fará a conexão para consultar se houve solicitação de bloqueio do veículo por parte do usuário da página web. Na Figura 12 visualiza-se o diagrama esquemático do hardware.

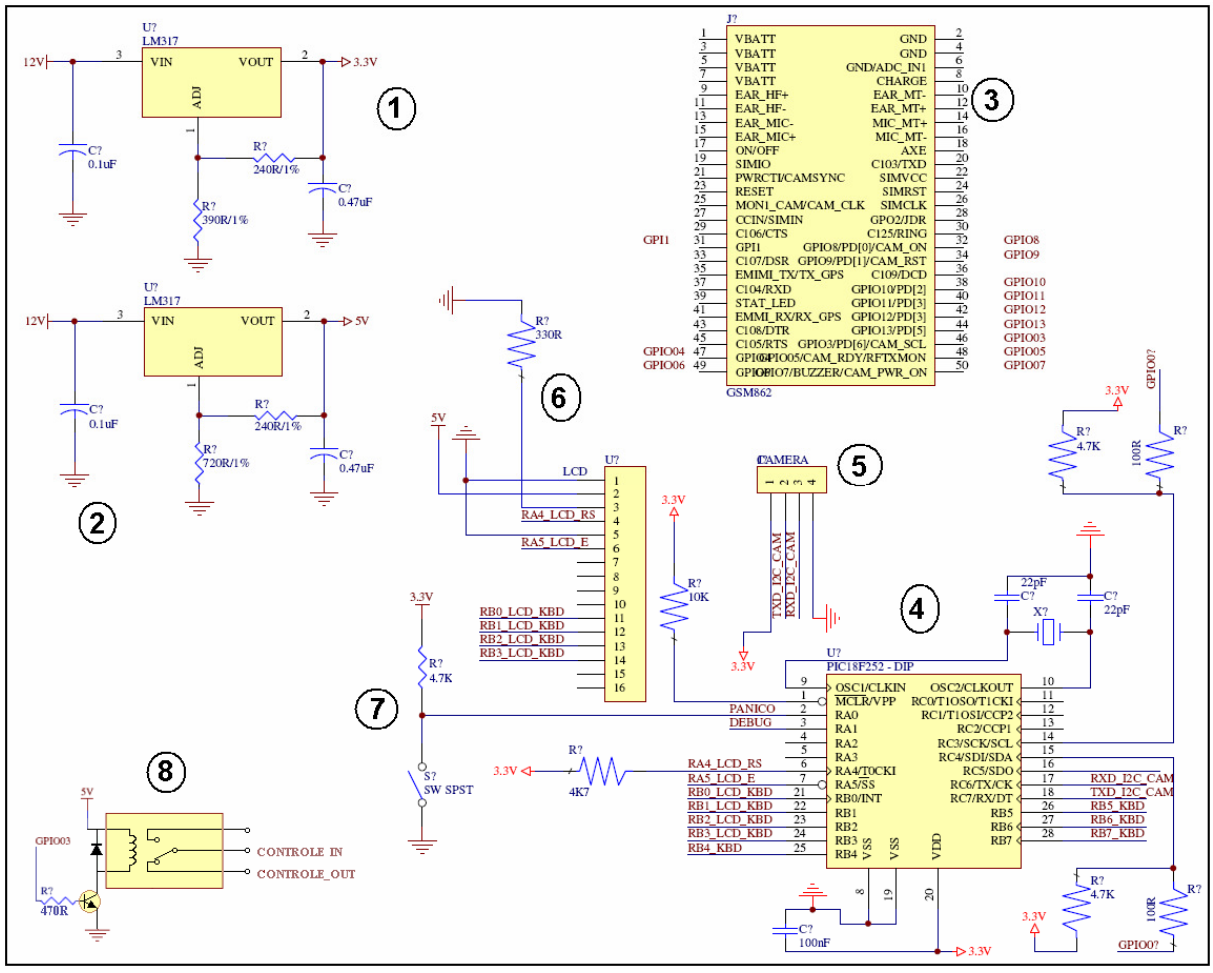


Figura 12 – Diagrama esquemático do hardware

As indicações numeradas no diagrama esquemático da Figura 12 destinam-se a

identificação de cada ligação.

Pode-se visualizar na indicação de número 1 e 2 os diagramas das ligações dos reguladores de tensão LM317 de 3.3 volts para o dispositivo escravo I2C e para alimentação da câmera, o outro LM317 é regulado para 5 volts para alimentação do display LCD.

A indicação de número 3 é apresentado o diagrama das conexões GPIO no modem GM862-GPS. As conexões de alimentação e demais são abstraídas, visto que o mesmo já é fornecido acoplado a uma placa de desenvolvimento do fabricante.

No indicador de número 4 são referenciados às ligações necessárias ao dispositivo escravo I2C, que pode ser identificado pelo microcontrolador PIC18F452, e sua conexão de alimentação ao LM317 regulado a 3.3 volts.

Na indicação de número 5 visualiza-se o diagrama da conexão da câmera I2C ao dispositivo escravo e sua alimentação ao LM317 regulado a 3.3 volts.

Na indicação de número 6 são detalhadas as conexões necessárias do display LCD e sua ligação ao dispositivo I2C assim como sua alimentação que é conectada ao LM317 regulado a 5.0 volts.

Na indicação numérica 7 é apresentado a conexão do sensor de pânico diretamente conectada ao dispositivo escravo I2C e alimentada pelo LM317 de 3.3 volts.

Finalmente no diagrama identificado com o número 8 podemos identificar a conexão para o relé de controle de bloqueio do veículo.

3.2.2 Software

Nas subseções seguintes, apresenta-se os diagramas relacionados ao desenvolvimento do software envolvido no sistema. A especificação do software embarcado, do software PC e da página web foi realizada utilizando-se os diagramas de Nassi-Schneiderman com o auxílio da ferramenta NSD-Editor (KALT, 1997).

3.2.2.1 Software embarcado

O software embarcado é o software contido no módulo Telit. Não há abordagem do software embarcado no dispositivo escravo I2C, visto que o mesmo foi desenvolvido por Basic4ever (2008) que cedeu a gravação do código no componente para utilização gratuita no

projeto, não liberando o código fonte para especificação do protocolo I2C no controle de acesso ao LCD, câmera serial e sensor de pânico.

Em relação ao software embarcado no dispositivo escravo, a empresa Basic4ever disponibilizou uma biblioteca desenvolvida na linguagem Python para livre acesso de todas as funções desenvolvidas no dispositivo escravo.

O software desenvolvido para o módulo Telit, nada mais é do que um *script*, criado e validado em um microcomputador pessoal. Esse *script* fará controle de todo o protótipo, direta ou quando indiretamente através dispositivo escravo I2C. O *script* aplicado ao módulo telit será responsável primeiramente pela leitura da localização, comunicação com o dispositivo escravo para acessar os periféricos, e enfim realizando uma conexão com o servidor e armazenando as informações relevantes ao sistema, conforme diagrama demonstrado na Figura 13.

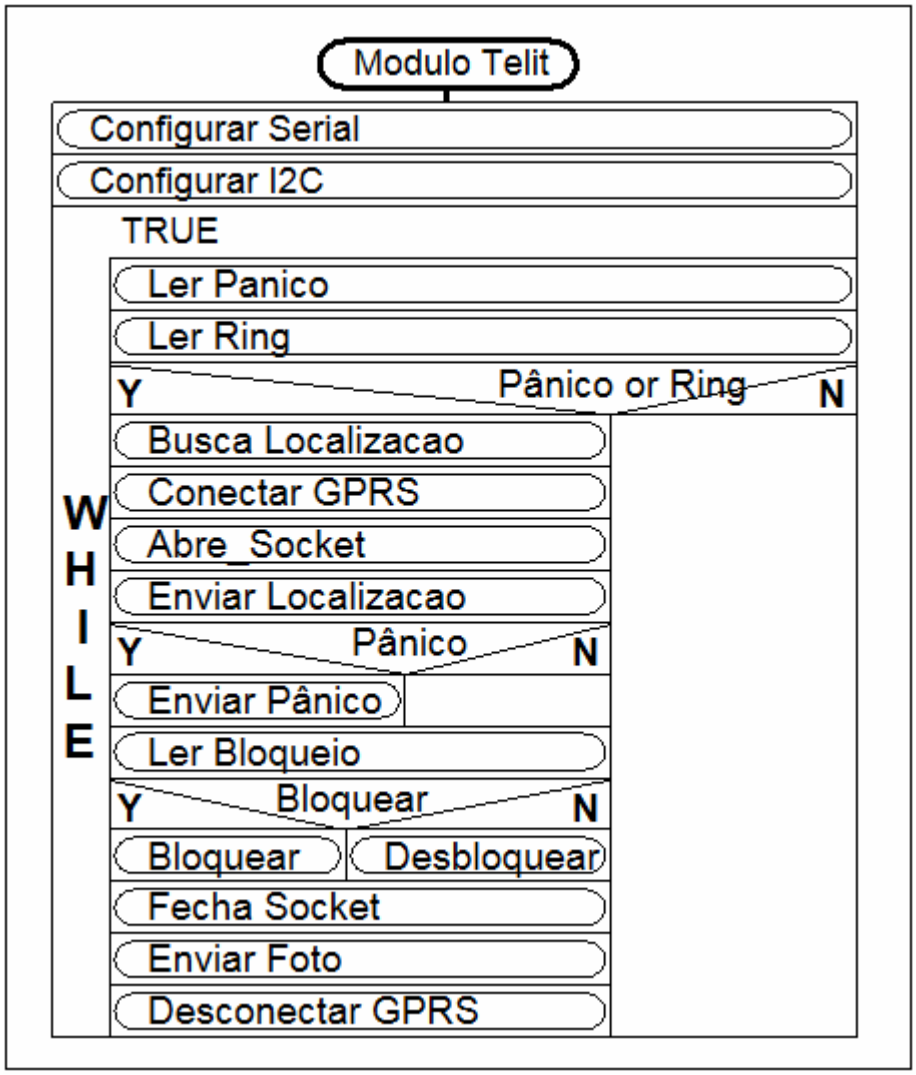


Figura 13 – Diagrama de Nassi-Schneiderman do módulo Telit

Ressalta-se que a Figura 14 indica a configuração de velocidade da porta de

comunicação serial e esta é utilizada no protótipo somente durante o período de desenvolvimento auxiliando na detecção e correção de falhas.

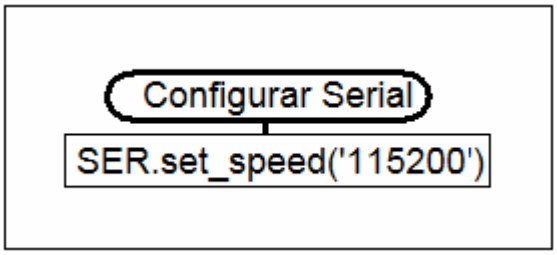


Figura 14 – Diagrama da rotina para configuração da comunicação serial

A inicialização do protocolo I2C se faz acionando a função própria disponibilizada na biblioteca de comunicação com o dispositivo escravo, esta definirá via software, quais pinos serão utilizados para o protocolo I2C, de acordo com a Figura 15.

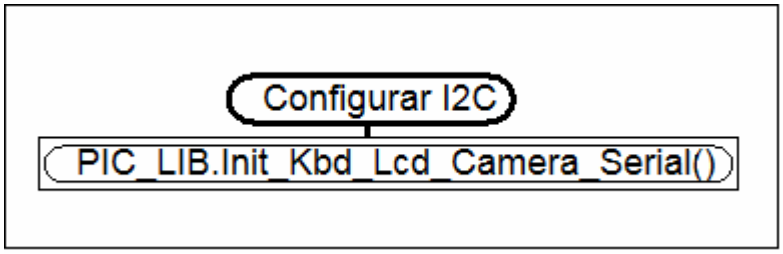


Figura 15 – Diagrama da rotina para inicialização da comunicação I2C através de biblioteca

Na Figura 16 faz-se a leitura do sensor de pânico utilizando biblioteca própria para comunicação com o dispositivo escravo. Essa leitura é essencial, visto que a detecção para situações de pânico deve eficiente possibilitando o registro rápido desta informação.

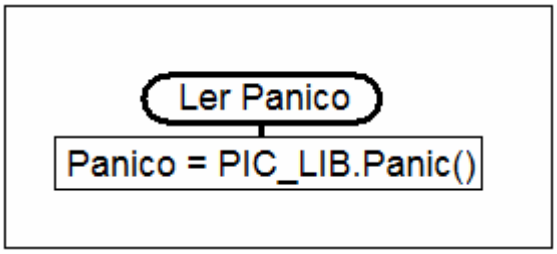


Figura 16 – Diagrama da rotina para leitura do botão pânico através de biblioteca

Na Figura 17 pode-se visualizar o diagrama para detecção de uma ligação de entrada, que no protótipo é utilizada como solicitação de conexão com o software PC. Esta função retornará a quantidade de chamadas detectadas pelo modem Telit.

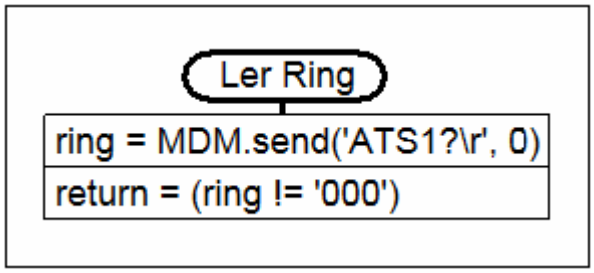


Figura 17 – Diagrama da rotina para detecção de ligação de entrada

O primeiro passo importante no protótipo é adquirir a posição do veículo, para isso utiliza-se a função disponibilizada pela biblioteca MDM que se comunica diretamente com o modem do protótipo e este com o módulo GPS buscando com isso a posição geográfica conforme Figura 18.

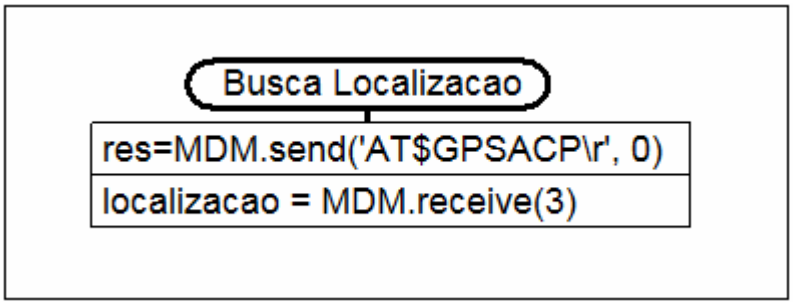


Figura 18 – Diagrama da rotina que busca localização

Para o envio das informações de localização, faz-se conexão GPRS (Figura 19), detalhada pelos comentários inseridos, possibilitando a abertura de um *socket* de comunicação (Figura 20), prosseguindo assim com o envio da informação, conforme Figura 21.

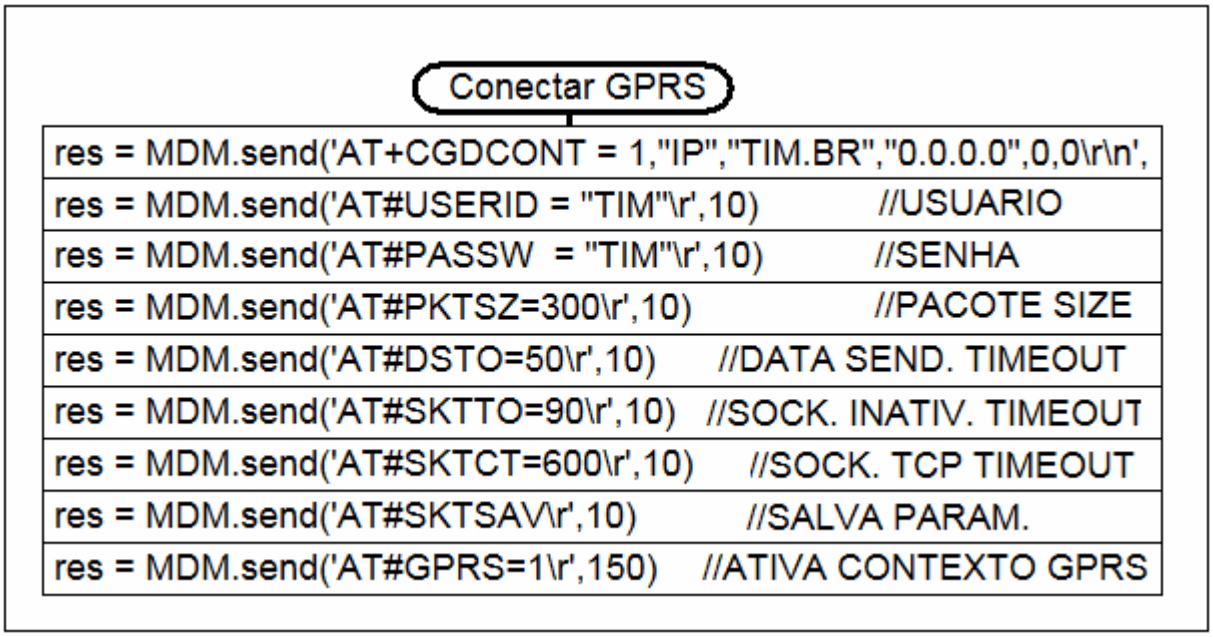


Figura 19 – Diagrama da rotina de conexão GPRS

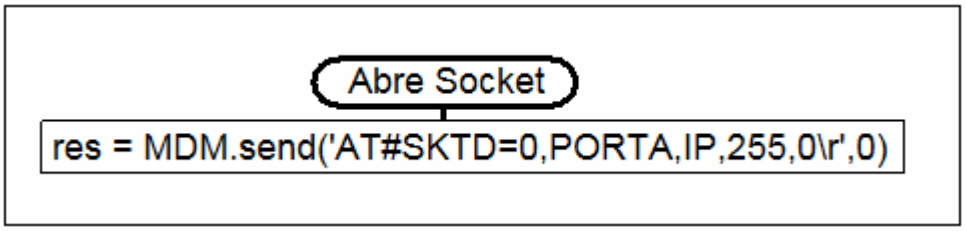


Figura 20 – Diagrama da rotina de abertura de *socket*

Após abertura do *socket*, envia-se a informação de localização a ser gravada, que será identificada e registrada pelo servidor da conexão *socket*. Na Figura 21 visualiza-se o envio dos dados.

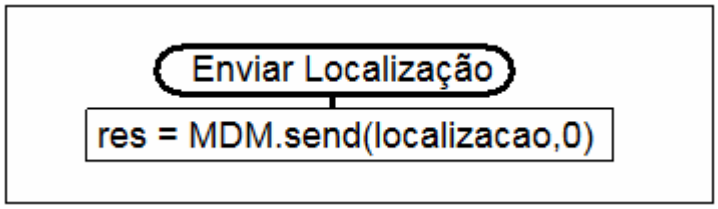


Figura 21 – Diagrama da rotina de envio da localização

Se a conexão está sendo ativada pelo acionamento do sensor de pânico, este aviso de pânico será registrado no servidor da conexão. Para isso é realizado o procedimento contido na Figura 22.

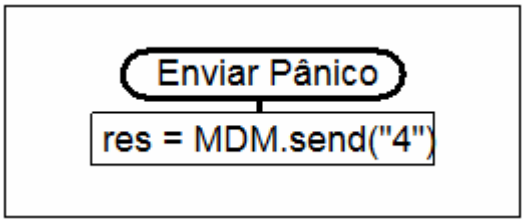


Figura 22 – Diagrama da rotina de envio para aviso de pânico

O próximo passo é realizar a consulta para verificar a necessidade do bloqueio do veículo. Para isso, ainda conectado via *socket* ao servidor, é enviado um parâmetro para o *socket*, aguardando assim o retorno indicando o bloqueio ou liberação do veículo, conforme Figura 23.

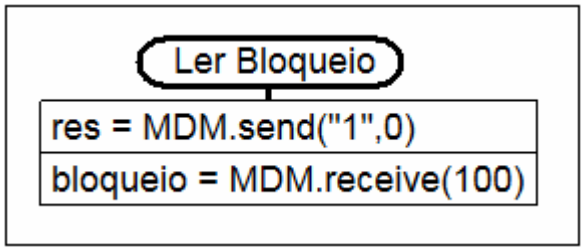


Figura 23 – Diagrama da rotina que realiza leitura do pedido de bloqueio

Depois de realizar a consulta de bloqueio do veículo, é realizado o bloqueio ou liberação do mesmo através das funções descritas nas Figuras Figura 23 e Figura 24.

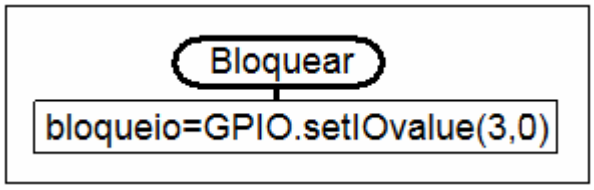


Figura 24 – Diagrama da rotina que efetua bloqueio do veículo

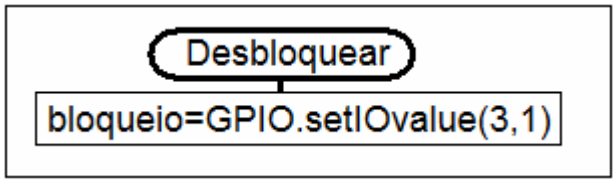


Figura 25 – Diagrama da rotina que efetua desbloqueio do veículo

Depois das operações necessárias realizadas via *socket*, este pode ser encerrado, de

acordo com o que pode ser visualizado na Figura 26.

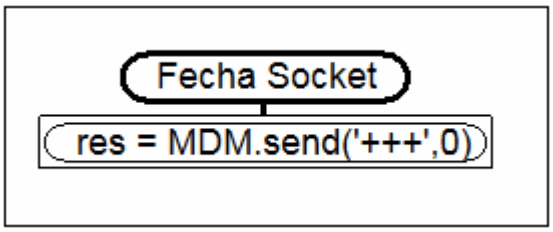


Figura 26 – Diagrama da rotina que fechamento do *socket*

Depois do socket encerrado, o protótipo realiza a captura de uma imagem do motorista do veículo. A função de captura de imagem é realizada pelo dispositivo escravo I2C. Se houver sucesso na captura de imagem, é iniciada a conexão FTP para transferência do arquivo para o computador servidor, de acordo com a Figura 27 e seguintes.

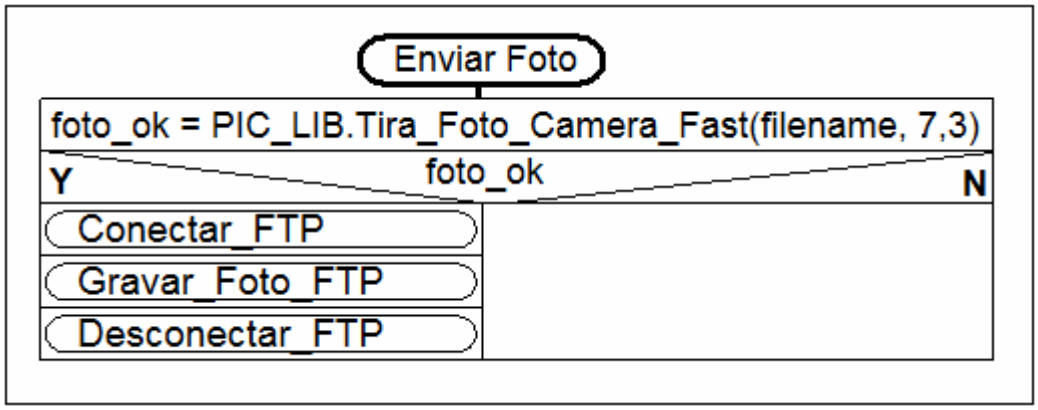


Figura 27 – Diagrama da rotina para captura e envio da imagem pelo protótipo

A operação de conexão de FTP é detalhada na figura abaixo, indicando os parâmetros necessários para realizar tal conexão.

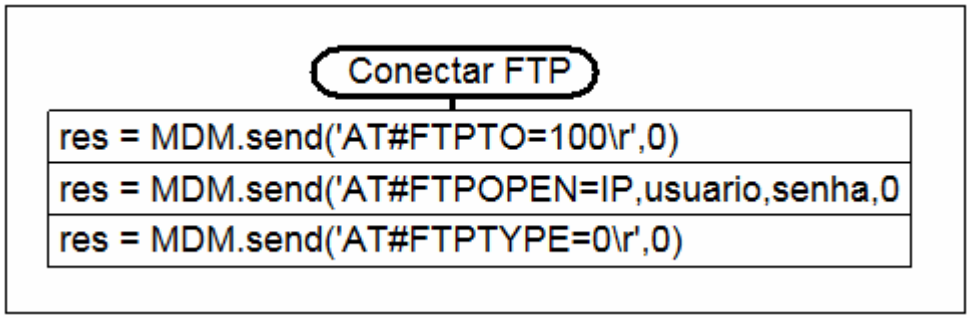


Figura 28 – Diagrama da rotina de conexão FTP

A operação de transferência do arquivo contendo imagem do motorista é detalhada na Figura 29. Nota-se neste diagrama a seqüência de comandos realizados durante a conexão FTP. Primeiramente define-se a criação do arquivo no servidor, em seguidas obtém-se o tamanho específico do arquivo, e com essa informação pode-se varrer a quantidade de bytes do arquivo e assim enviar os bytes que compõem a imagem desejada, encerrando o arquivo com a indicação dessa operação ao modem com os caracteres '+++'.

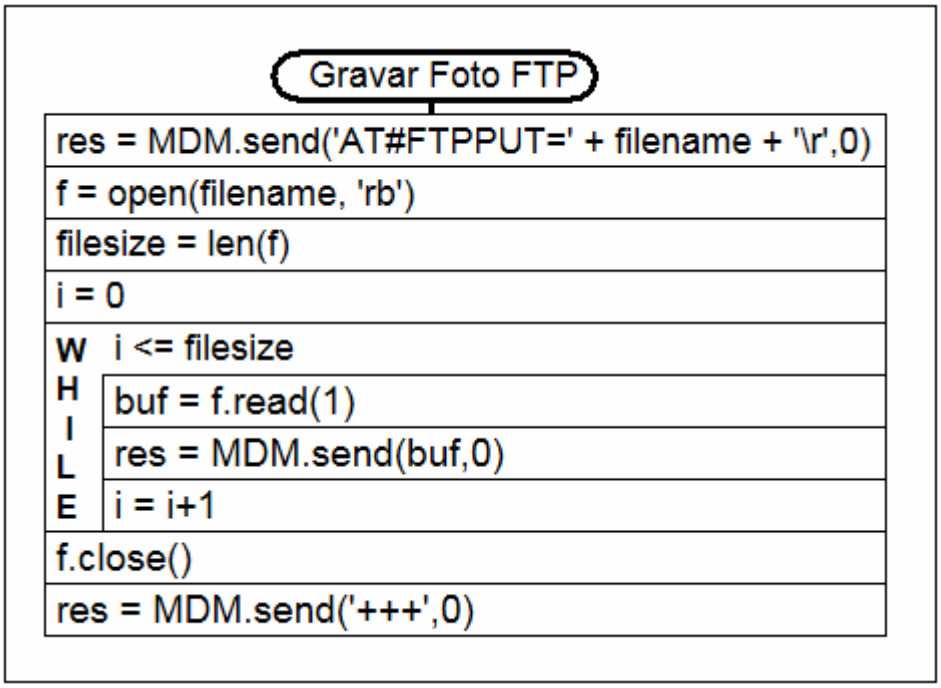


Figura 29 – Diagrama da rotina de gravação da imagem no FTP

Depois de realizado a transferência do arquivo, é realizado o fechamento da conexão FTP e da conexão GPRS, encerrando o processo de gravação das informações pelo protótipo, conforme Figura 30 e Figura 31.

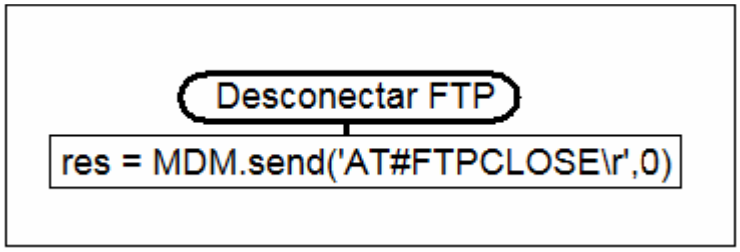


Figura 30 – Diagrama da rotina de desconexão FTP

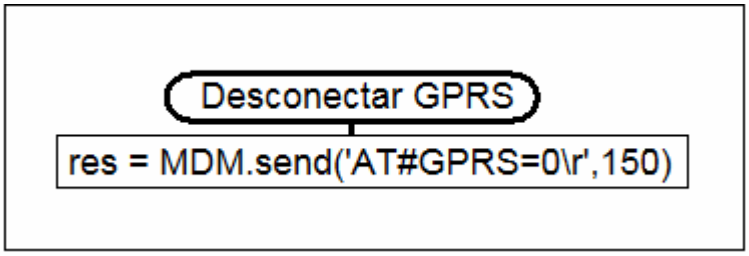


Figura 31 – Diagrama da rotina de desconexão GPRS

3.2.2.2 Software PC

Desenvolveu-se este software para possibilitar a comunicação direta do protótipo com o computador servidor a fim de possibilitar o registro das informações desejadas, assim

também como permitir a consulta da base de dados por parte do protótipo, visto que o as solicitações de bloqueio ou desbloqueio do veículo são realizadas pelo usuário via página web de consulta.

Na Figura abaixo pode-se visualizar a especificação do bloco principal de execução do software PC.

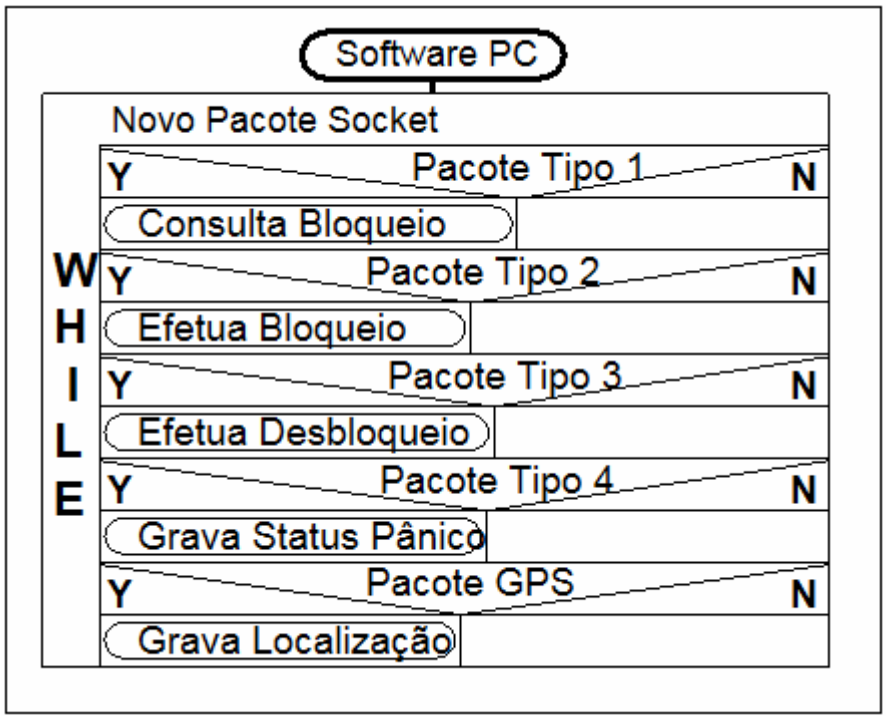


Figura 32 – Visualização do bloco principal de execução do software PC

As rotinas citadas na figura anterior não serão detalhadas nesta seção, visto que apenas realizam consultas ou gravação de informação única na base de dados.

Para visualização detalhada destas rotinas de acesso à base de dados, vide seção de implementação do software PC.

3.2.2.3 Software de consulta

Foi desenvolvida em PHP uma página web para realizar as consultas diretamente no banco de dados SQLServer. A comunicação com a base de dados é de vital importância para o software de consulta, pois ela deverá disponibilizar as informações mais recentes sobre a localização e estado do veículo fornecidas pelo equipamento.

Outra função de destaque para a página será de possibilitar a visualização de um aviso gráfico em tela, indicando estado de pânico no veículo. Este sinal de pânico será acionado através de um botão localizado no interior do veículo próprio para este fim.

A partir disso, o usuário poderá realizar o pedido de bloqueio do veículo, fazendo uso de um botão disponível na página web.

Haverá também a visualização do estado real do veículo, visto que o pedido de bloqueio pode ser efetuado pelo usuário da página, mas somente depois que o protótipo realizar a conexão ao servidor e confirmar o pedido de bloqueio, é que realmente fará o bloqueio do veículo.

Na Figura 33 pode-se visualizar as funções disponibilizadas ao usuário através da página web de consulta.

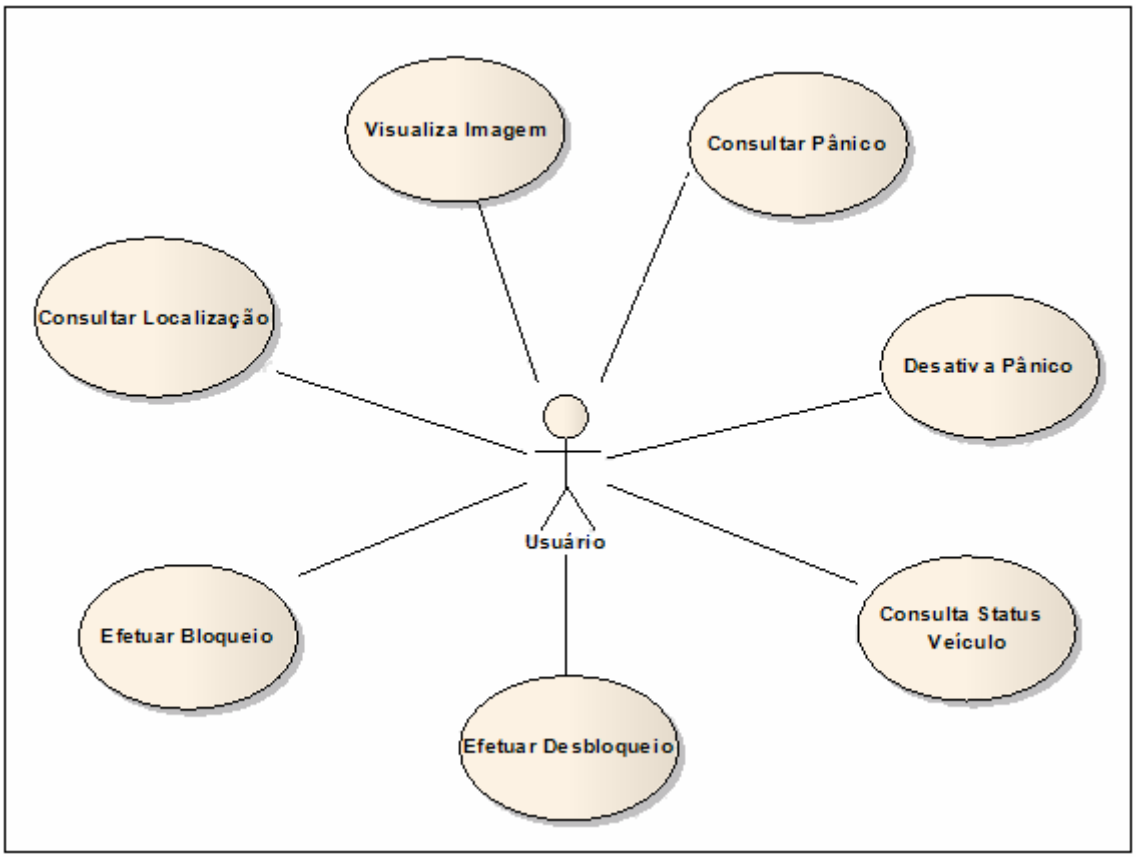


Figura 33 – Diagrama de caso de uso para software de consulta

Para adicionar maior detalhamento sobre as operações realizadas pelo usuário pode-se visualizar a Figura 34 que demonstra através de um diagrama de seqüência o funcionamento do software de consulta.

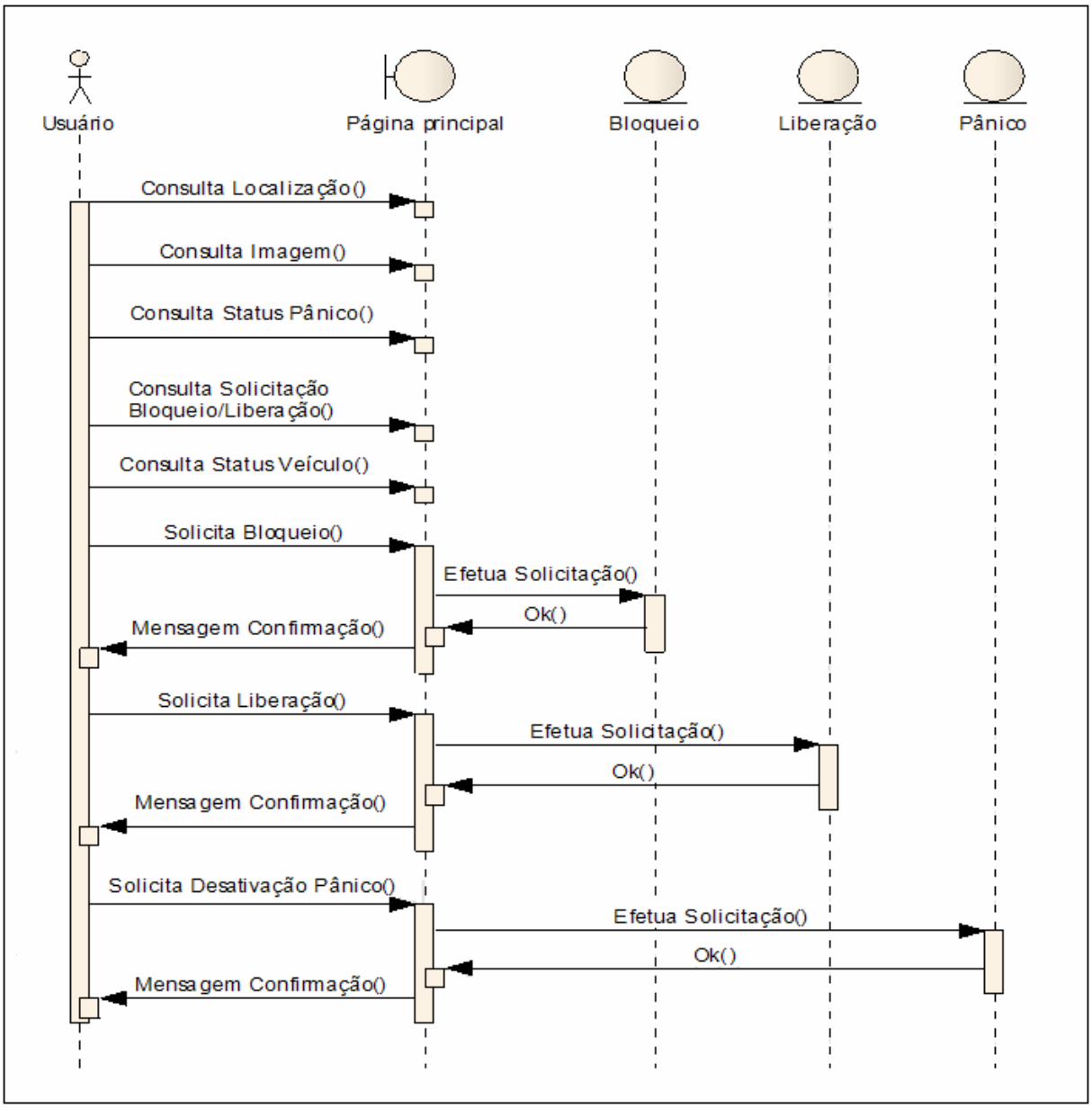


Figura 34 – Diagrama de seqüência para software de consulta

3.3 IMPLEMENTAÇÃO

A seguir é apresentada a implementação, mostrando as técnicas e ferramentas utilizadas na confecção do hardware e software além da operacionalidade da implementação.

3.3.1 Técnicas e ferramentas utilizadas

Para a implementação do software embarcado no módulo Telit, foi utilizada linguagem Python com o ambiente de desenvolvimento PythonWin, utilizando programação estruturada, conforme especificam os diagramas de Nassi-Schneiderman.

Também utilizou-se a ferramenta Boland Delphi 5.0 para realizar a implementação do software PC, que tem como principal objetivo aguardar as conexões efetuadas pelo protótipo e assim receber as informações enviadas pelo mesmo.

Para auxílio no desenvolvimento do software embarcado, fez-se uso também da ferramenta de comunicação serial Hyper Terminal versão 5.1 disponibilizada no sistema operacional Windows XP Professional. O Hyper Terminal foi utilizado com a configuração 115200 bits/s, 8 bits de dados, N de paridade, 1 para bits de parada e N para controle de fluxo, para então ser conectado diretamente ao kit de desenvolvimento Telit através de um cabo USB para visualizar as mensagens inseridas ao longo do código de *script* desenvolvido facilitando a detecção de erros durante o processo.

Já no software de consulta, pagina web, utilizou-se a linguagem PHP, de acordo com a especificação no diagrama do software de consulta.

3.3.2 Hardware

Para a implementação do hardware foi utilizado o módulo Telit GM862-GPS juntamente com um kit de desenvolvimento Telit, agregando em uma só placa muitos dos requisitos exigidos na especificação como controle de portas de entrada e saída a fim de prover controle sobre acionamento de dispositivos de segurança do veículo e suporte a linguagem de programação Python.

Abaixo na Figura 35 pode visualizar a placa de desenvolvimento e sua pinagem de acesso às entradas/saídas do módulo Telit.

Figura 35 – Kit de desenvolvimento Telit

Adicionou-se também ao projeto uma placa universal para montagem padrão de circuitos na medida de 10cm x 20cm. A placa proporcionou fixação do dispositivo escravo I2C e também dos demais periféricos como a conexão com a câmera e o display LCD.

O centro do protótipo é a placa de desenvolvimento Telit, ela é conectada ao dispositivo escravo através de uma ligação de comunicação I2C. O dispositivo escravo, por sua vez, possui conexão de comunicação com a câmera, com o display LCD e também com o sensor de pânico.

A fonte de alimentação é externa à placa. Esta alimentação de 12volts divide-se uma via para alimentação direta da placa de desenvolvimento Telit. Outra via desta fonte, é utilizada para alimentação do dispositivo escravo com 3.3 volts, câmera com 3.3 volts e display LCD utilizando 5 volts.

Para estabilizar as duas tensões de 3.3 e 5 volts, fez-se uso de dois reguladores LM317. Estes componentes tem como principal objetivo ajustar a tensão de cada componente eletrônico, por isso estão presentes na grande maioria dos dispositivos eletrônicos desenvolvidos.

Para o ajuste das tensões citadas, utilizaram-se resistores variáveis, R2 (Figura 36), até que na saída do LM317 houvesse a tensão desejada. As ligações necessárias para estes

componentes podem ser visualizadas na Figura 36.

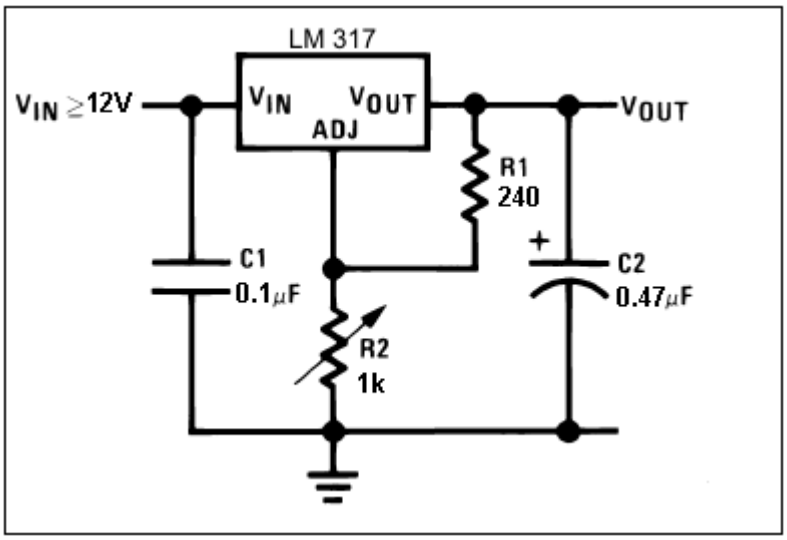


Figura 36 – Ajuste de tensão no LM317

Finalmente abaixo na Figura 37, visualiza-se a montagem final do protótipo desenvolvido.

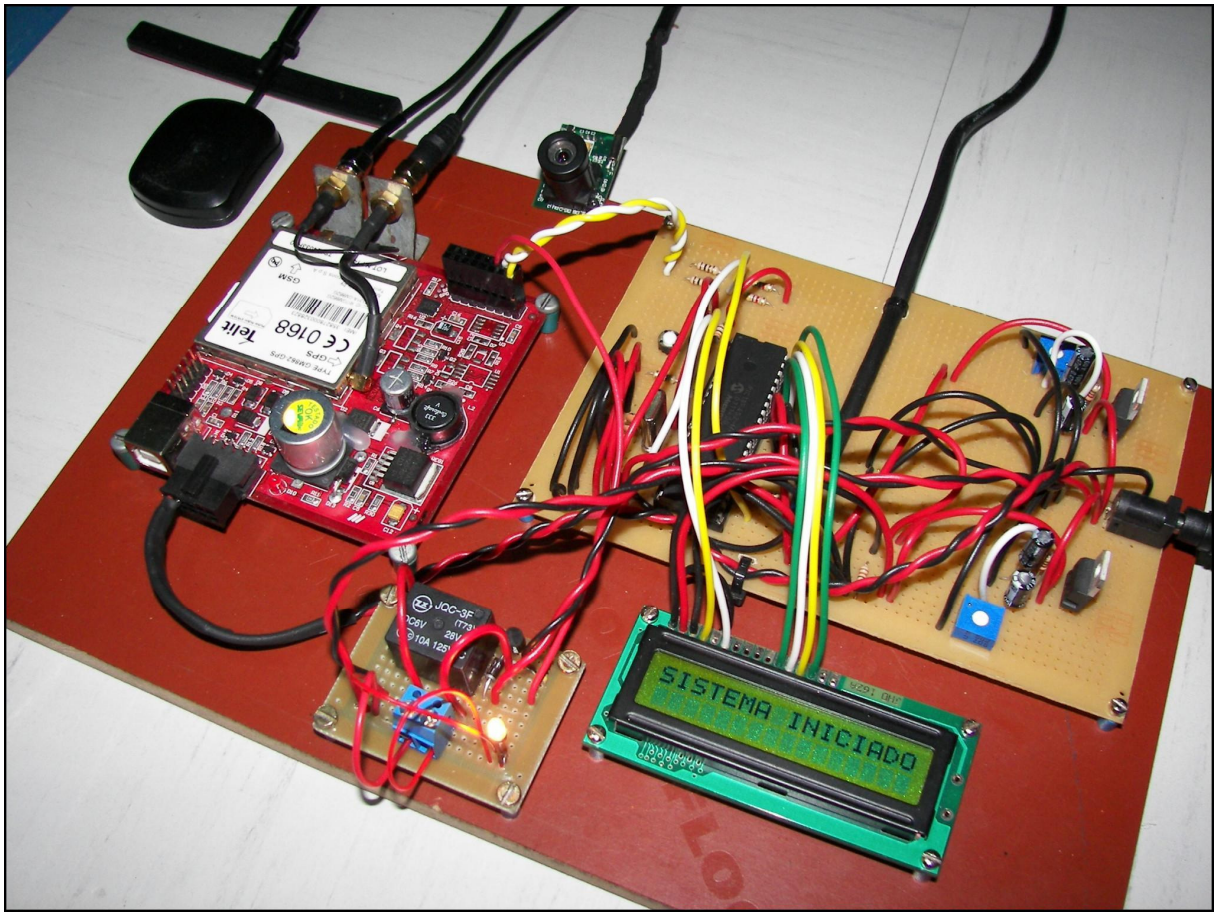


Figura 37 – Visualização do protótipo desenvolvido

3.3.3 Software

A implementação dos softwares envolvidos no desenvolvimento do protótipo são apresentados nas subseções seguintes, seguindo as especificações realizadas em capítulo específico.

3.3.3.1 Software embarcado

Para o software embarcado no protótipo, primeiramente se fez necessário o conhecimento das bibliotecas nativas e disponíveis no módulo Telit. Essas bibliotecas permitem a utilização direta dos recursos do módulo, bem como as portas de comunicação da placa de desenvolvimento Telit.

Na implementação do software embarcado desenvolvido, pode-se notar que as bibliotecas disponíveis do módulo Telit são facilmente acionadas e utilizadas no projeto, assim também como a biblioteca fornecida por Basic4ever (2008) para utilização das funções desenvolvidas no dispositivo escravo, conforme Quadro 2.

```
import MOD
import IIC
import MDM
import GPIO
import SER
import PIC_LIB
...
def Config_IIC():
    PIC_LIB.Init_Kbd_Lcd_Camera_Serial()
...
```

Quadro 2 – Visualização da sintaxe de uso de bibliotecas em Python

É importante destacar que o protótipo se destina primeiramente a disponibilizar a localização do veículo. Para reduzir custos indevidos, faz-se a transmissão das informações apenas quando solicitado pelo usuário. A solicitação pode ser efetuada pelo sensor de pânico ou quando o módulo recebe uma chamada pelo número GSM registrado. O Quadro 3 demonstra função para leitura do sensor de pânico assim também como o Quadro 4 ilustra a função para detecção de chamada recebida.

```
panico = PIC_LIB.Panic()
```

Quadro 3 – Visualização da função de leitura do sensor de pânico

```
def Ring():
    res = MDM.send('ATS1?\r', 0)        #Qtd rings detectados
    res = MDM.receive(5)
    if (res.find('OK') != -1):
        tmp = res.split("\r\n")
        res = tmp[1]
    else:
        res = '000'

    if (res == '000'):                  #Indica sem ligação
        res = 0
    else:                               #Indica ligação de entrada
        res = MDM.send('ATH\r',0)       #Finaliza Ligação de entrada
        res = MDM.receive(10)
        MOD.sleep(50)
        res = 1
    return res
```

Quadro 4 – Visualização da função para detectar chamada recebida

O Quadro 5 demonstra a utilização de função específica para aquisição das coordenadas de localização do veículo através da biblioteca de comunicação direta com o modem.

```
def Busca_Localizacao():
    res = MDM.send('AT$GPSACP\r', 0)            #Solicita localizacao
    res = MDM.receive(3)
    if (res.find('OK') != -1):
        tmp = res.split("\r\n")                 #retira espaços
        res = tmp[1]
        tmp = res.split(" ")
        res = tmp[0] + " " + tmp[1]
    else:
        res = ""
    return res
```

Quadro 5 – Visualização de comando para solicitar posicionamento geográfico

Depois de adquirir as informações de localização, é efetuada a conexão para enviar estas informações ao computador servidor. A conexão é realizada via socket, após configurar conexão GPRS, conforme Quadro 6.

```
def Conecta_GPRS():
    Send_Lcd_1("TENTANDO CONEXAO")
    res = MDM.send('AT#GPRS=0\r',0)
    res = MDM.receive(150)
    res = MDM.send('AT+CGDCONT = 1,"IP","TIM.BR","0.0.0.0",0,0\r\n',10)
    res = MDM.receive(50)                              #inicia Conexao
    res = res.find('OK')
    if (res != -1):
        res = MDM.send('AT#USERID = "TIM"\r',0)     #usuario
        res = MDM.receive(10)
        res = MDM.send('AT#PASSW  = "TIM"\r',0)     #senha
        res = MDM.receive(10)
        res = MDM.send('AT#PKTSZ=300\r',0)          #tamanho pacote
        res = MDM.receive(10)
        res = MDM.send('AT#DSTO=50\r',0)            #timeout conexao
        res = MDM.receive(10)
        res = MDM.send('AT#SKTTO=90\r',0)           #timeout socket
        res = MDM.receive(10)
        res = MDM.send('AT#SKTCT=600\r',0)          #pacote tcp
        res = MDM.receive(10)
        res = MDM.send('AT#SKTSAV\r',0)             #salva parametros
        res = MDM.receive(10)
        res = MDM.send('AT#GPRS=1\r',0)             #inicia GPRS
        res = MDM.receive(150)
        res = res.find ('+IP')
        if (res != -1):
            Send_Lcd_1("CONEXAO GPRS OK ")
            res = "OK"
        else:
            res = "FALHOU"
    else:
        res = "FALHOU"
    return res
```

Quadro 6 – Visualização da configuração para conexão GPRS

Após conexão GPRS iniciada, faz-se uso do estabelecimento de um *socket* com o microcomputador servidor, criando um canal para comunicação de acordo com Quadro 7.

```
def Conecta_Socket():
    Send_Lcd_1("ABRINDO SOCKET  ")
    res = MDM.send('AT#SKTD=0,PORTA,IP,255,0\r',0) #abre socket
    res = MDM.receive(50)
    res = res.find ('CONNECT')
    if (res != -1):
        Send_Lcd_1("SOCKET OK       ")
        res = "OK"
    else:
        res = "FALHOU"
    return res
```

Quadro 7 – Visualização do estabelecimento de *socket* para comunicação

Para o envio da imagem capturada pelo protótipo, é realizada conexão FTP. Facilitando assim a manipulação do arquivo de imagem.

O FTP é iniciado de acordo com código anexado ao Quadro 8.

```
def Conecta_Ftp():
    Send_Lcd_1("ABRINDO FTP     ")
    res = MDM.send('AT#FTPCLOSE\r',0)
    res = MDM.receive(250)
    res = MDM.send('AT#FTPTO=100\r',0)                    #timeout ftp
    res = MDM.receive(50)
    res = MDM.send('AT#FTPOPEN=IP,USUARIO,SENHA,0\r',0) #inicia ftp
    res = MDM.receive(250)
    res = res.find('OK')
    if (res != -1):
        Send_Lcd_1("FTP OK          ")
        res = MDM.send('AT#FTPTYPE=0\r',0)            #tipo arq. binario
        res = MDM.receive(50)
        res = MDM.send('AT#FTPCWD="public_html"\r',0) #acessa diretorio
        res = MDM.receive(80)
        res = MDM.send('AT#FTPCWD="fotos"\r',0)       #acessa diretorio
        res = MDM.receive(80)
        res = "OK"
    else:
        res = "FALHOU"
    return res
```

Quadro 8 – Visualização da conexão FTP para transferência de arquivo

Depois da conexão FTP estabelecida, é feita abertura do arquivo de imagem e a transferência do conteúdo do arquivo, ou seja, os bytes que compõem a imagem capturada. Os detalhes desta operação podem ser visualizados no Quadro 9.

```
def Grava_Foto_Ftp(filename):
    Send_Lcd_1("GRAVAR FOTO     ")
    res = MDM.send('AT#FTPPUT=' + filename + '\r',0)   #cria arquivo
    res = MDM.receive(100)
    res = res.find('CONNECT')
    if (res != -1):
        try:
            arquivo = "FOTO.JPG"
            f = open(arquivo, 'rb')                     #abre arquivo bin
            f.seek(0,2)                                 #final arquivo
            tam = f.tell()                              #posicao atual
            f.seek(0)                                   #inicio arquivo
            Send_Lcd_1("TAMANHO: " + str(tam))
            res = MOD.sleep(20)
            i = 0
            while (i <= tam):
                buf = f.read(500)                   #le 500 bytes arquivo
                res = MDM.send(buf,0)               #envia buffer
                i = i+500
                Send_Lcd_2(str(i))
                MOD.sleep(5)
            f.close()                               #fecha arquivo
            Send_Lcd_2("                ")
            MOD.sleep(30)
        finally:
            Finaliza_Foto()
            res = "OK"
    else:
        res = "FALHOU"
    return res
```

Quadro 9 – Visualização do processo de transferência de arquivo via FTP

Destacando a funcionalidade da biblioteca para uso do dispositivo escravo e seus periféricos, nota-se através do Quadro 10 que a comunicação se torna extremamente simples.

```
def Send_Lcd_1(msg):
    PIC_LIB.Send_Lcd(chr(254) + chr(128) + msg + chr(0))    #ESCREVE LINHA
    res = MOD.sleep(10)
```

Quadro 10 – Visualização da utilização de biblioteca para comunicação com dispositivo escravo

Assim também como no Quadro 11 abaixo, pode-se visualizar a função específica para solicitar foto ao dispositivo escravo.

```
def Tira_Foto():
    Send_Lcd_1("CAPTURA FOTO    ")
    nomearquivo = "FOTO.JPG"
    dummy = PIC_LIB.Tira_Foto_Camera_Fast(nomearquivo,7,3)
    return dummy
```

Quadro 11 – Visualização de função para captura de imagem

O controle de bloqueio do veículo pode ser acionado com um simples sinal de acionamento do relé, emitido por uma porta de uso geral do protótipo, conforme Quadro 12.

```
def Liga_Rele():
    r=GPIO.setIOvalue(3,1)            #liga rele

def Desliga_Rele():
    r=GPIO.setIOvalue(3,0)            #desliga rele
```

Quadro 12 – Visualização de função para controle de acionamento do relé

O bloco principal de execução do software embarcado é exibido no Quadro 13.

```
...
while 1:
    panico = PIC_LIB.Panic() #ord(panico) == 0 aciona panico
    ring = Ring()
    if (ord(panico) == 0) or (ring == 1):
        localizacao = Busca_Localizacao()
        res = localizacao.find('GPSACP:')
        if (res != -1):
            res = Conecta_GPRS()
            res = res.find('OK')
            if (res != -1):
                try:
                    res = Conecta_Socket()
                    res = res.find('OK')
                    if (res != -1):
                        try:
                            if (ord(panico) == 0):
                                res = MDM.send("4",0)   #grava panico
                                res = MDM.receive(100)
                            res = MDM.send(localizacao,0)
                            nomefoto = MDM.receive(100)
                            Send_Lcd_Clr(str(nomefoto))
                            res = MDM.send("1",0)
                            bloquear = MDM.receive(100) #consulta bloqueio
                            if (bloquear == "1"):
                                Send_Lcd_1("BLOQUEAR VEICULO")
                                Desliga_Rele()
                                res = MDM.send("2",0)   #conf. bloqueio
                                res = res.find("OK")
                            else:
                                Send_Lcd_1("VEICULO LIBERADO")
                                Liga_Rele()
                                res = MDM.send("3",0)   #conf. liberacao
                                res = res.find("OK")
                        finally:
                            res = Desconecta_Socket()
                        foto_ok = Tira_Foto()
                        if (foto_ok == 0):
                            res = Conecta_Ftp()
                            res = res.find("OK")
                            if (res != -1):
                                try:
                                    Grava_Foto_Ftp(nomefoto)
                                finally:
                                    Desconecta_Ftp()
                finally:
                  res = Desconecta_GPRS()
        PIC_LIB.Reset_Pic()
```

Quadro 13 – Visualização do bloco principal de execução do software embarcado

3.3.3.2 Software PC

O software desenvolvido para o PC é de vital importância para a o protótipo. O software PC disponibilizará a base de dados para o registro das informações recebidas, assim também como realizar os pedidos de consulta efetuados pelo protótipo.

Este software foi desenvolvido em Delphi, e tem como principal função aguardar as conexões que o protótipo pode iniciar. O protótipo instalado no veículo poderá conectar

diretamente via *socket* de comunicação disponibilizado pelo computador servidor e depois da conexão estabelecida, realizar as operações de acordo com os parâmetros enviados pelo protótipo no veículo.

No quadro abaixo visualiza-se as funções que o software PC poderá realizar, de acordo com parâmetros recebidos via conexão *socket*.

```
procedure TForm_GPS.ServerSocket1ClientRead(Sender: TObject;
Socket: TCustomWinSocket);
var
  S1, S2: string;  // Lê bytes da conexão

...

begin
  S1 := Socket.ReceiveText;         //RECEBE CONTEUDO

  if (S1 = '1') then                //CONSULTA BLOQUEIO
    Consulta_Bloqueio
  else
  if (S1 = '2') then                //EFETUA BLOQUEIO DO VEICULO
    Bloqueia_Veiculo
  else
  if (S1 = '3') then                //EFETUA DESBLOQUEIO DO VEICULO
    Desbloqueia_Veiculo
  else
  if (S1 = '4') then                //GRAVA STATUS PANICO
    Grava_Panico
  else
  if (pos('$GPSACP', S1) > 0) then  //GRAVA LOCALIZACAO, RETORNA NOME FOTO
    Grava_Localizacao;
end;
```

Quadro 14 – Método principal para recebimento de pacote do *socket*

Os quadros a seguir detalham as funções demonstradas no Quadro 14.

```
function Consulta_Bloqueio : string;
begin
  with qry_temp do
  begin
    close;
    sql.clear;
    sql.add('SELECT STATUS_BLOQUEIO FROM STATUS');
    open;
    result := IntToStr(Fields[0].AsInteger);
    close;
  end;
end;
```

Quadro 15 – Função para consultar solicitação de bloqueio

```
function Bloqueia_Veiculo : string;
begin
  with qry_temp do
  begin
    close;
    sql.clear;
    sql.add('UPDATE STATUS SET STATUS_VEICULO = 1');
    ExecSQL;
    if qry_temp.RowsAffected > 0 then
      result := 'OK'
    else
      result := 'FALHOU';
  end;
end;
```

Quadro 16 – Função para gravar bloqueio do veículo

```
function Desbloqueia_Veiculo : string;
begin
  with qry_temp do
  begin
    close;
    sql.clear;
    sql.add('UPDATE STATUS SET STATUS_VEICULO = 0');
    ExecSQL;
    if qry_temp.RowsAffected > 0 then
      result := 'OK'
    else
      result := 'FALHOU';
  end;
end;
```

Quadro 17 – Função para gravar desbloqueio do veículo

```
function Grava_Panico : string;
begin
  with qry_temp do
  begin
    close;
    sql.clear;
    sql.add('UPDATE STATUS SET STATUS_PANICO = 1');
    ExecSQL;
    if qry_temp.RowsAffected > 0 then
      result := 'OK'
    else
      result := 'FALHOU';
  end;
end;
```

Quadro 18 – Função para gravar pânico no veículo

Para realizar a extração correta das informações de localização no desenvolvimento do protótipo necessita-se analisar a informação de localização recebida. Esta informação recebida é idêntica à obtida pelo protótipo ao pedido de localização para o módulo GPS.

A informação recebida pelo software PC é na verdade uma palavra que contem todas as informações de localização separadas por vírgula conforme Quadro 19.

```
$GPSACP:173243.999,2654.2223S,04905.9671W,2.2,39.7,3,47.04,0.10,0.05,280508,04

$GPSACP: – Indicação de informação de localização;
173243.999 – Informação da hora (17:32:43);
2654.2223 – Informação da latitude (26°54.2223’);
04905.9671 – Informação da longitude (049°05.9671’);
2.2 – Informação de precisão;
39.7 – Informação de altitude;
3 – Informação de dimensão (2 ou 3);
47.04 – Informação de curso do planeta;
0.10 – Informação de velocidade km/h;
0.05 – Informação de velocidade knots;
280508 – Informação de data (28/05/08);
04 – Informação da quantidade de satélites detectados;
```

Quadro 19 – Parâmetro recebido contendo informações de localização

```
function Grava_Localizacao : string;

...

    try
      AUX_MSG := LOCAL_MENSAGEM;

      delete(AUX_MSG, 1, Pos(':', AUX_MSG));      //Apaga $GPSACP:
      while pos(' ', AUX_MSG) > 0 do              //Tira espaços
        delete(AUX_MSG, pos(' ', AUX_MSG), 1);
      AUX_HORA := Copy(AUX_MSG, 1, 6);            //Copia Hora = 125959
      delete(AUX_MSG, 1, pos(',', AUX_MSG));      //Apaga Hora;
      LOCAL_LATITUDE := Copy(AUX_MSG, 1, pos(',', AUX_MSG) - 1);
      delete(AUX_MSG, 1, pos(',', AUX_MSG));      //Apaga Latitude;
      LOCAL_LONGITUDE := Copy(AUX_MSG, 1, pos(',', AUX_MSG) - 1);
      delete(AUX_MSG, 1, pos(',', AUX_MSG));      //Apaga Longitude;
      delete(AUX_MSG, 1, pos(',', AUX_MSG));      //Apaga Precisao;
      delete(AUX_MSG, 1, pos(',', AUX_MSG));      //Apaga Altitude;
      delete(AUX_MSG, 1, pos(',', AUX_MSG));      //Apaga Dimensao 2/3D;
      delete(AUX_MSG, 1, pos(',', AUX_MSG));      //Apaga Curso Terra;
      LOCAL_VELOCIDADE := Copy(AUX_MSG, 1, pos(',', AUX_MSG) - 1);
      delete(AUX_MSG, 1, pos(',', AUX_MSG));      //Apaga Veloc. em Km;

      //LIMPAR COORDENADAS RETIRANDO INDICACAO PONTO CARDEAL (N,S,E,W)
      delete(LOCAL_LATITUDE, length(LOCAL_LATITUDE), 1);
      delete(LOCAL_LONGITUDE, length(LOCAL_LONGITUDE), 1);

      //ACRESCENTAR GRAUS E MINUTOS-26°54.2223',-049°05.9671'
      LOCAL_LATITUDE := Copy(LOCAL_LATITUDE, 1, 3) + '°' +
                        Copy(LOCAL_LATITUDE, 4, 7) + #39;
      LOCAL_LONGITUDE := Copy(LOCAL_LONGITUDE, 1, 4) + '°' +
                         Copy(LOCAL_LONGITUDE, 5, 7) + #39;

      //Copia Data = 010108 e adiciona hora
      AUX_DATA := Copy(AUX_MSG, 1, 6) + AUX_HORA;

      // NOME DA FOTO SERÁ A DATA  + HORA
      LOCAL_FOTO := AUX_DATA + '.jpg';

      // FORMATA DATA + HORA 010108125959 -> 01/01/2008 12:59:59
      LOCAL_DATA := copy(AUX_DATA,1,2) + '/' +
                    Copy(AUX_DATA,3,2) + '/' +
                    '20' + Copy(AUX_DATA,5,2) + ' ' +
                    copy(AUX_DATA,7,2) + ':' +
                    Copy(AUX_DATA,9,2) + ':' +
                    Copy(AUX_DATA,11,2);


      ...
      //REALIZA INSERÇÃO NA BASE DE DADOS
      ...

    except
      result := 'FALHOU';
    end;

...
```

Quadro 20 – Função para gravar localização do veículo

3.3.3.3 Software de consulta

O software de consulta, ou página web, foi desenvolvido em PHP permitindo assim

consultas diretas na base de dados do computador servidor, informando visualmente ao usuário a coordenadas de localização do veículo. Na página web serão também disponibilizados dois ícones de atalho ao lado de cada registro de localização. O primeiro fará acesso a um mapa virtual para facilitar a visualização da posição do veículo. O segundo fará a visualização da imagem capturada pela câmera instalada no veículo.

Alem destas facilidades citadas, o software de consulta realizará também ações de bloqueio e desbloqueio do veículo com o acionamento de simples botões devidamente identificados e localizados na parte superior da página. O bloqueio será realizado por funções próprias desenvolvidas em php de acordo com visualização do Quadro 21 e Quadro 22.

```
<?

...

{
    $_QUERY ="UPDATE STATUS SET STATUS_BLOQUEIO = 1";
    $res=mssql_query($_QUERY);
}


echo "<script language=javascript>
   <!–
   alert('Bloqueio Solicitado com Sucesso!');
   parent.window.location = 'index.php';
   //-->
   </script>";

?>
```

Quadro 21 – Visualização da função que realiza pedido de bloqueio do veículo

```
<?

...

{
    $_QUERY ="UPDATE STATUS SET STATUS_BLOQUEIO = 0";
    $res=mssql_query($_QUERY);
}

echo "<script language=javascript>
   <!–
   alert('Desbloqueio Solicitado com Sucesso!');
   parent.window.location = 'index.php';
   //-->
   </script>";

?>
```

Quadro 22 – Visualização da função que realiza pedido de desbloqueio do veículo

O software de consulta também disponibilizará painéis informando o status atual do veículo, como “Pânico” quando o sensor de pânico é acionado, “Bloqueio do Veículo Solicitado”, “Desbloqueio do Veículo Solicitado”, “Veículo Bloqueado” e “Veículo Desbloqueado”, todos estes, indicam as possíveis situações de acordo com a interação do motorista com o sensor de pânico instalado no veículo e do usuário às facilidades disponibilizadas pelo software de consulta.

A página principal exibirá as informações registradas na base de dados pelo protótipo. Estas informações serão disponibilizadas em forma de tabela, facilitando a identificação de cada informação. Esta operação pode ser detalhadamente visualizada através do Quadro 23.

```
...

echo"<table width=90%>";
echo"<tr>";
echo"<td><p>Data/Hora</p></td>";
echo"<td><p>Latitude</p></td>";
echo"<td><p>Longitude</p></td>";
echo"<td><p>Velocidade</p></td>";
echo"<td><p>Mapa</p></td>";
echo"<td><p>Foto</p></td>";

$_QUERY ="SELECT * FROM LOCALIZACAO ORDER BY LOCAL_DATA DESC";
$res=mssql_query($_QUERY);

while($rec=mssql_fetch_array($res))
{
   $i^=0x01;
   if ($i)
      echo"<tr CLASS=row_1 >";
   else
      echo"<tr CLASS=row_2 >";

   echo"<td><p>" . $rec[LOCAL_DATA] . "</p></td>";
   echo"<td><p>" . $rec[LOCAL_LATITUDE] . "</p></td>";
   echo"<td><p>" . $rec[LOCAL_LONGITUDE] . "</p></td>";
   echo"<td><p>" . $rec[LOCAL_VELOCIDADE] . "Km/h</p></td>";

   $message="Localização do Protótipo";
   echo"<td align=\"center\"><a href=\"http://maps.google.com.br/maps?
q=$rec[LOCAL_LATITUDE],+$rec[LOCAL_LONGITUDE]+($message)\"><img
ALT=\"GOOGLE MAPS\" border=0 src=\"world.gif\" width='48px'
height='48px'></a></td>";

   echo"<td align=\"center\"><a
href=\"http://www.visys.com.br/~leandro/fotos/$rec[LOCAL_FOTO]\"><img
ALT=\"FOTO\" border=0
src=\http://www.visys.com.br/~leandro/fotos/$rec[LOCAL_FOTO]\ width='64px'
height='48px'></a></td>";

...
```

Quadro 23 – Visualização do código principal desenvolvido para o software de consulta

3.3.4 Operacionalidade da implementação

Uma empresa, proprietária de um veículo de serviço, deseja realizar o monitoramento e controle da localização do seu veículo, que é utilizado por vários funcionários durante o dia e noite. Além da localização, a empresa deseja também monitorar a ocupação no veículo através de imagens capturadas por uma câmera no interior do mesmo.

A empresa poderá, via página web, realizar o bloqueio e desbloqueio do veículo remotamente, em qualquer situação, principalmente quando em uma ocorrência não prevista,

o motorista tenha acionado o botão de pânico no interior do veículo, indicando um possível alerta por roubo ou seqüestro.

3.3.4.1 Instalação

O equipamento deverá ser acondicionado no interior do veículo, em um local seguro, a fim de evitar contato com qualquer intempérie. Para a alimentação do protótipo, poderá ser utilizar a fonte de energia do próprio veículo, no caso a bateria 12 volts que pode ser conectada ao protótipo através de uma conexão direta à bateria utilizando um fusível de no máximo 5 ampéres, justificados pela especificação do fabricante do módulo telit adicionando o consumo do dispositivo escravo, câmera serial e do relé de controle, assegurando eficiência contra qualquer possível curto-circuito.

Deverá também se desejado, realizar a conexão elétrica do protótipo com qualquer dispositivo eletrônico que possa prover bloqueio do veículo, neste caso, será utilizado a ligação com a alimentação da bobina de ignição do veículo.

Na Figura 38 visualiza-se as conexões necessárias de alimentação do protótipo e do relé de corte da bobina de ignição.

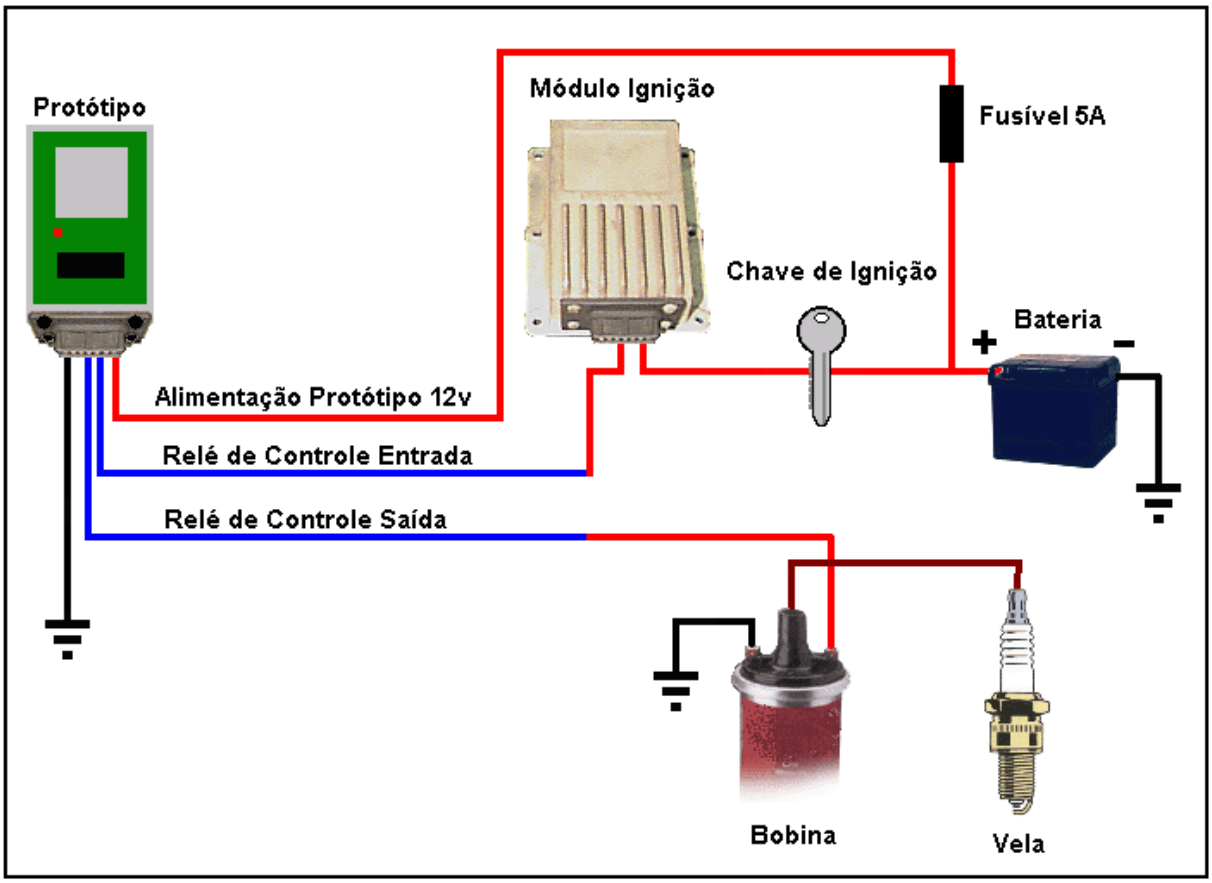


Figura 38 – Esquema de ligação do protótipo

As antenas também deverão ser conectadas ao protótipo e bem fixadas ao veículo, assim também como a câmera para captura de imagens.

A instalação do software de consulta, a página web, poderá ser realizada em qualquer servidor, pois será utilizado apenas para consulta das informações de localização. O servidor deverá dispor de uma base de dados SQLServer para proporcionar consulta das informações registradas tanto pela página web quanto o software PC.

Será também necessário a instalação do software PC, que estará aguardando conexões em um endereço IP fixo e porta específica, previamente programadas no protótipo para conexão e registro das informações de localização.

3.3.4.2 Monitoramento e controle

O monitoramento iniciará após a alimentação do protótipo, visto que a página deverá estar previamente instalada, assim como o software PC, para então receber as informações que o protótipo fornecer.

Para proporcionar comodidade e economia, os dados serão enviados somente quando o

protótipo receber uma ligação em seu modem para o número registrado no cartão da operadora GSM ou quando o sensor de pânico for ativado, indicando assim uma situação atípica. Este sinal indica que o usuário deseja que o protótipo conecte ao servidor e grave as informações, do contrário, não terá controle das conexões necessárias do protótipo, indicando assim custos elevados e inviabilidade de uso. Na Figura 39 visualiza-se o software PC em comunicação.

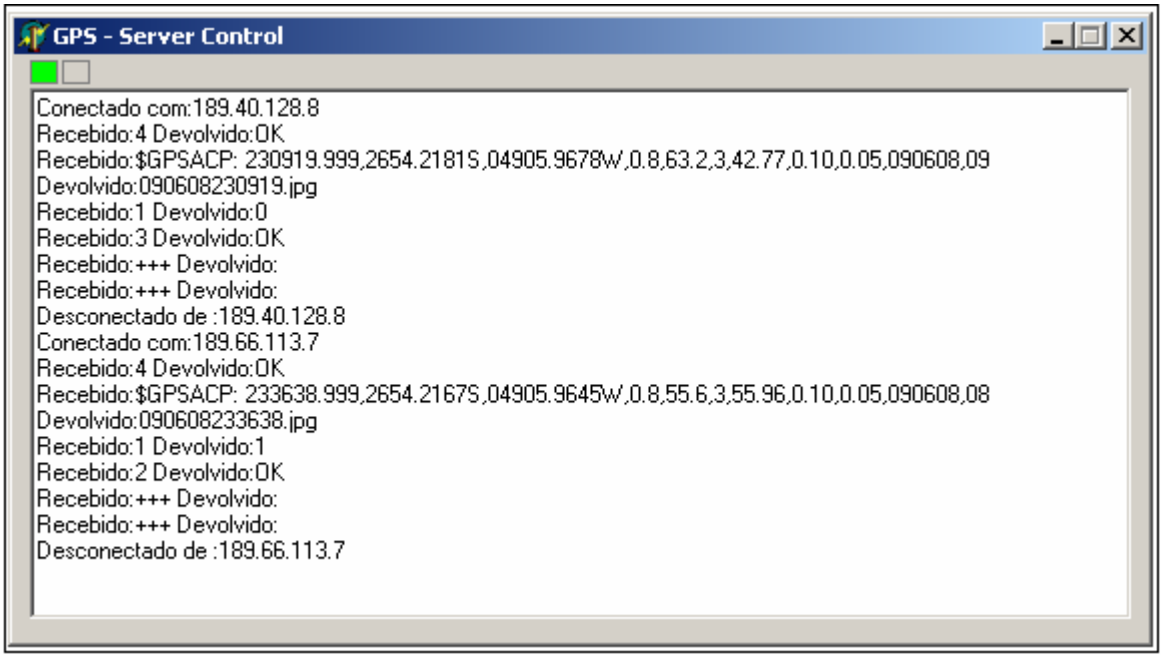


Figura 39 – Visualização do software PC e comunicação

O usuário poderá, com as informações analisadas na página web, realizar o bloqueio ou liberação do veículo, bastando para isso registrar esse pedido na base de dados e novamente, realizar uma chamada para o protótipo para que ele consulte esta base de dados e realize a tarefa solicitada.

Uma demonstração desta situação pode ser analisada mediante Figura 40, onde pode-se perceber que o veículo encontra-se bloqueado e em estado de pânico. Pode-se visualizar também os ações que podem ser efetuadas, como a desativação do estado de pânico, assim também como solicitar a liberação do veículo com um simples clique do mouse sobre a operação desejada.

Data/Hora	Latitude	Longitude	Velocidade	Mapa	Foto
Jul 07 2008 09:23AM	-26°54.2804'	-049°04.8976'	0.36Km/h		
Jul 06 2008 03:52PM	-26°50.4626'	-048°37.8631'	38.26Km/h		
Jul 06 2008 03:30PM	-26°49.7529'	-048°37.4788'	32.22Km/h		
Jul 06 2008 03:26PM	-26°49.9709'	-048°37.6438'	37.98Km/h		
Jul 06 2008 03:15PM	-26°53.0389'	-048°38.5445'	36.68Km/h		
Jul 06 2008 03:12PM	-26°53.7768'	-048°39.1455'	34.70Km/h		
Jul 06 2008 02:54PM	-26°52.1453'	-048°38.3233'	46.69Km/h		

Figura 40 – Visualização do software de consulta

Na Figura 40 pode-se também notar as informações de localização obtidas do veículo, permitindo inclusive visualização sobre um mapa (Figura 41) facilitando a identificação da localização do veículo.

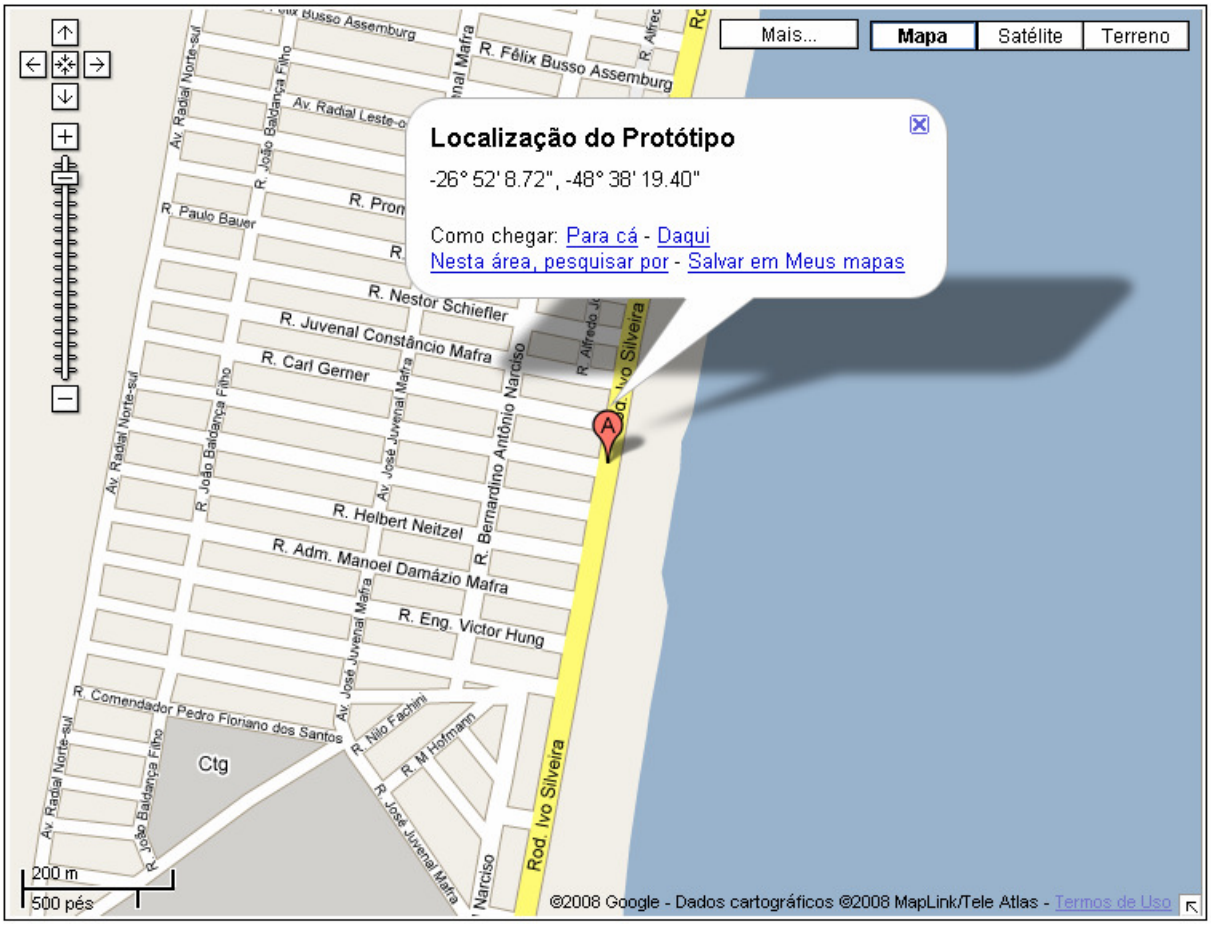


Figura 41 – Visualização do mapa de localização do protótipo

Uma miniatura da imagem capturada também é disponibilizada para rápida identificação do local ou do motorista e pode ser maximizada (Figura 42 e Figura 43) clicando-se com o mouse sobre a miniatura desejada.

Figura 42 – Visualização de imagem do local capturada pelo protótipo

Figura 43 – Visualização de imagem do motorista capturada pelo protótipo

3.4 RESULTADOS E DISCUSSÃO

Desenvolver um protótipo de rastreamento de veículos foi considerado uma tarefa complexa, devido à quantidade de dispositivos envolvidos, além das diferentes tecnologias de cada um destes. A dificuldade está na integração da comunicação entre eles.

A utilização do dispositivo escravo I2C para o controle dos periféricos adicionados ao protótipo como a câmera, o LCD e o sensor de pânico, se mostrou totalmente eficiente.

É importante ressaltar que, em relação aos trabalhos correlatos, há destaque no desenvolvimento deste protótipo, visto que possibilitou a integração total entre a implementação realizada em hardware e as implementações em software, tanto para o software PC quanto para a página web, principalmente pelas diferentes linguagens adotadas em cada uma das implementações.

No Quadro 24 é apresentado o comparativo das características do protótipo

desenvolvido com os trabalhos correlatos.

Características	**Protótipo desenvolvido**	**Hasegawa (1999)**	**Weigang (2001)**	**Belório (2005)**
Implementação em Software	*	*	*	
Implementação em Hardware	*			
Captura de imagem	*			
Visualização via web	*		*	*
Tempo Real		*	*	*
Comunicação	GSM	Serial	GSM	Satélite Banda C
Linguagem de Programação	Python, PHP e Delphi	C	JAVA	Não Informado

Quadro 24 – Características do protótipo desenvolvido e trabalhos correlatos

4 CONCLUSÕES

Os objetivos inicialmente traçados foram atendidos de forma total. O principal objetivo, que era fazer a integração completa das diferentes tecnologias adotadas no sistema, se deu como principal fator de satisfação.

A possibilidade de bloquear e desbloquear o veículo à distância, pela página web, adicionam uma incrível facilidade e praticidade no monitoramento e controle do veículo pelo usuário.

A utilização do dispositivo escravo foi determinante para a conclusão deste trabalho. Seria inviável a construção de um equipamento limitando a visualização apenas das coordenadas geográficas de GPS. Assim, através deste dispositivo, é possível fazer o monitoramento visual do motorista do veículo através da câmera além de proporcionar controle eletrônico para dispositivos adicionais como o LCD.

Durante a implementação do software embarcado foi possível observar as principais particularidades da programação e manipulação direta em hardware, obrigando até a trabalhar algumas informações *byte* a *byte* para proporcionar os resultados desejados no protótipo.

O sistema possui algumas limitações já conhecidas nas aplicações que utilizam comunicação via satélite para o GPS e via antena para o GSM, onde a localização interfere diretamente na recepção dos sinais. O GPS, por exemplo, não funciona dentro de túneis ou prédios, assim como no GSM pode haver perda de sinal de acordo com a localização das antenas disponibilizadas pela operadora GSM.

Uma limitação que não pode ser ignorada é de que o protótipo foi desenvolvido inicialmente para controle de apenas um veículo. Mas é importante destacar que os softwares desenvolvidos são facilmente extensíveis com pequenas alterações.

Outra dificuldade encontrada foi referente à placa de desenvolvimento Telit fabricado por Seva (2008), que não forneceu documentação necessária do produto adquirido. Todo o desenvolvimento do hardware foi prejudicado em seu cronograma, visto que a falta de documentação do produto exigiu testes específicos na pinagem do equipamento, a fim de descobrir a que se destinava efetivamente cada pino disponível.

Acionando também como dificuldade encontrada, cita-se a de não localizar câmera serial disponível à venda no Brasil. Foi necessário realizar aquisição do componente através de importação direta dos Estados Unidos.

4.1 EXTENSÕES

Utilizar a disponibilidade e gratuidade das informações de localização GPS para implementar um mapa eletrônico de navegação do veículo.

Utilizar as entradas e saídas extras do dispositivo escravo para expandir recursos e controlar vários elementos eletrônicos do veículo com um único equipamento.

Possibilitar bloqueio e liberação do veículo através de SMS recebido quando não há possibilidade de acesso à página web.

Expandir o controle via web dos equipamentos conectados ao protótipo.

Realizar controle de acesso para usuários da página web, a fim de implementar privacidade e controle individual de bloqueio para cada equipamento.

REFERÊNCIAS BIBLIOGRÁFICAS

BASIC4EVER. [S.l.], 2008. Disponível em: <http://www.basic4ever.com>. Acesso em: 10 jan. 2008.

BELÓRIO, Cristiano Leles. **Descrição de um sistema de rastreamento veicular utilizando GPS.** 2005. 51 f. Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) - Faculdade de Ciências Aplicadas de Minas, Uberlândia, MG.

CÂMARA, Gilberto. **Perspectivas ao norte do equador.** Savannah, out. 2000. Disponível em: <http://www.giscience.org/giscience2000/camara.html>. Acesso em: 20 set. 2007.

CATUNDA, Marco. **Python**: guia de consulta rápida. São Paulo: Novatec, 2001. 128 p.

DRAGO, Daniele; DISPERATI, Attilio Antonio. **Aspectos básicos sobre GPS.** Curitiba: FUPEF, 1996. 16 p.

HAMMOND, Mark. **PythonWin**: help. Version 1.5.2. [S.l.], 1999. Documento eletrônico disponibilizado com o pacote Telit Python 1.5.2.

HASEGAWA, Júlio Kiyoshi et al. Sistema de localização e navegação apoiado por GPS. In: CONGRESSO BRASILEIRO DE CARTOGRAFIA, 20., 1999, Recife-PE. **Anais...** CD-ROM.

KALT, Marcel. **NSD-Editor**: Nassi-Shneiderman diagram editor. [S.l.], 1997. Disponível em: <http://diuf.unifr.ch/softeng/student-projects/completed/kalt/ftp-nsd.html>. Acesso em: 10 maio 2008.

KONDO, Márcia N. S. et al. Estudo de requisitos do software embarcado no segmento da telemedicina. In: CONGRESSO BRASILEIRO DE INFORMÁTICA EM SAÚDE., Florianópolis, SC, 2006. **Anais...** São Paulo: SIBIS, 2006. p.1171-1176..

LUTZ, Mark. **Programming Python**.2nd ed. Beijing; S : astopol, CA : O´Reilly, c2001. xxxvii, 1255 p, il. +, 1CD-ROM.

MONTIBELLER JUNIOR, Ariberto. **Protótipo de sistema de monitoramento remoto utilizando TCP/IP sobre Ethernet (802.3).** 2005. 65 f. Trabalho de conclusão de curso - Universidade Regional de Blumenau, Curso de Ciências da Computação, Blumenau. Disponível em: <http://www.bc.furb.br/docs/MO/2005/305466_1_1.pdf>. Acesso em: 20 set. 2007.

PÉRICAS, Francisco Adell. **Redes de computadores:** conceitos e a arquitetura internet. Blumenau: Edifurb, 2003.

ROCHA, Cézar Henrique Barra. **GPS de navegação:** para mapeadores, trilheiros e navegadores. Juiz de Fora, MG: Universidade Federal de Juiz de Fora, 2003. 124 p.

SANTIS, Daniele. **SxPyDownloadTool**. [S.l.], 2006. Disponível em: <http://www.areasx.com/index.php?D=1&page=articoli.php&id=8169>. Acesso em: 17 mar. 2008.

SEIXAS FILHO, Constantino. **Comunicação através de sockets sobre TCP/IP**. Belo Horizonte, 2007. Disponível em: <http://www.cpdee.ufmg.br/~seixas/PaginaSDA/Download/SDADownload.htm>. Acesso em: 25 set. 2007.

SEVA. **Engenharia**. [S.l.], 2007. Disponível em: < http://www.seva.com.br/institucional.aspx >. Acesso em: 20 set. 2007.

SILVA, Abraão Balbino e; MOREIRA, Luiz Roberto Borges. **Telemetria em sistemas de comunicação móvel celular.** 2005. 141 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) - Faculdade de Tecnologia, Universidade de Brasília, Brasília.

SILVA, Pedro; PINHO, Felipe. **Controle e desenvolvimento de mini-submarino**. [S.l.], 2007. Disponível em: <http://ave.dee.isep.ipp.pt/~1000150/images/Imagens%20submarino/1000150_1010639rel_final.pdf>. Acesso em: 11 maio 2008.

SVERZUT, José Umberto. **Redes GSM, GPRS, EDGE e UMTS**: evolução a caminho da terceira geração (3G). São Paulo: Érica, 2005. 454 p.

TAURION, Cezar. **Software embarcado**: oportunidades e potencial de mercado. Rio de Janeiro: Brasport, 2005.

TELIT. **GM862 product description**. [S.l.], 2007. Disponível em: <http://www.telit.com/en/products/gsm-gprs.php?p=7&p_ac=show>. Acesso em: 26 set. 2007.

UCPROS. **Synchronous microcontroller communication interfaces**: SPI, Microwire and I2C Protocol Formats. [S.l.], 2008. Disponível em: <http://www.ucpros.com/work%20samples/Microcontroller%20Communication%20Interfaces%202.htm>. Acesso em: 21 maio 2008.

WEIGANG, Li et al. Implementação do sistema de mapeamento de uma linha de ônibus para um sistema de transporte inteligente. In: SEMINÁRIO INTEGRADO DE SOFTWARE E HARDWARE (SEMISH), 2001, Fortaleza. **Anais...** XXI Congresso da Sociedade Brasileira de Computação. v. 1. p. 72-85.

WORLDTIMEZONE. **GSM World Coverage Map and GSM Country List**. [S.l.], 2008. Disponível em: <http://www.worldtimezone.com/gsm.html>. Acesso em: 11 março 2008.