UNICEUB – CENTRO UNIVERSITÁRIO DE BRASÍLIA
FAET – FACULDADE DE CIÊNCIAS EXATAS E TECNOLOGIA
CURSO DE ENGENHARIA DA COMPUTAÇÃO

GUSTAVO DE FARIA SILVA

MONITORAMENTO DE AMBIENTE RESIDENCIAL COM USO DE CÂMERA E ACIONAMENTO DE ALARME

BRASÍLIA/DF
2º SEMESTRE DE 2007

GUSTAVO DE FARIA SILVA

MONITORAMENTO DE AMBIENTE RESIDENCIAL COM USO DE CÂMERA E ACIONAMENTO DE ALARME

Monografia apresentada ao Curso de Engenharia da Computação, como requisito parcial para obtenção do grau de Engenheiro de Computação.

Orientador: Prof. Aderlon M. Queiroz

BRASÍLIA/DF

2º SEMESTRE DE 2007

Resumo

Neste projeto é apresentada a construção de um sistema de hardware e software para o monitoramento de alarme residencial por meio de um telefone celular. É utilizada uma câmera estática que focaliza o ambiente a ser monitorado, sendo efetuada análise das imagens adquiridas para detectar a presença de um possível intruso. Quando da presença de um intruso, o mecanismo aciona o alarme e envia uma mensagem ao telefone celular. Desta forma, é possível acessar a página e ver a foto do intruso no momento que o mesmo invadiu a residência. Ainda, é possível ativar e desativar o alarme por meio do telefone celular acessando a internet com o mesmo.

Palavras-Chave: Detecção de Movimento, Monitoração de Ambientes, Análise de Imagem, Sistemas de Vigilância.

Abstract

In this project it is presented the construction of a hardware and software system for the monitor of residential alarm through a cellular phone. It is used a static camera that focalizes the place to be monitored, analyzing the acquired images to detect a possible intruder's presence. When of an intruder's presence, the mechanism works the alarm and it sends a message to the cellular telephone. This way, it is possible to access the web page and to see the intruder's picture in the moment that the intruder's picture invaded the residence. Still, it is possible to activate and to disable the alarm through the cellular telephone accessing the web page.

Keywords: Movement Detection, Environment Monitoring, Image Analysis, Surveillance System.

Agradecimentos

Agradecer é algo que enriquece o ser humano. Por isso, primeiramente eu agradeço a Deus por tudo que ele me proporcionou. Tudo mesmo, pois é nos momentos mais difíceis que nos preparamos e fortalecemos para enfrentarmos as diversas dificuldades que a vida nos traz.

Em segundo lugar, eu agradeço aos meus pais por terem me dado a devida educação e o apoio quando eu sempre precisei. Obrigado pai (Divino Pedro da Silva) e mãe (Lúcia de Faria Silva) por também saberem dizer não quando foi preciso.

Obrigado tia (Maria Eunice) por me acolher em sua casa e me dar força e apoio todos esses anos. Sou profundamente grato por tudo.

Também não poderia esquecer de você, Edgar Vidal, que desde o início criou as devidas condições para que eu pudesse conseguir êxito em mais essa caminhada.

Agradeço também ao meu avô (Faria) e a meu falecido tio (Euripedes) pelo exemplo de vida que vocês foram e sempre serão para mim.

Ainda, não poderia esquecer dos meus amigos e colegas de faculdade: Michel Calheiros, Rodrigo Benin “Xamba”, Daniel Neto, Roberta Neder, Vagner Bastos e Davi Venâncio. Obrigado a todos vocês.

Agradeço também ao pessoal do Exército que acompanharam e fizeram parte dessa caminhada: Cel Papa, Cel Luciano, Cel Veras, Cap Herculano, Ten Paula Pacheco, Ten Bassan, Ten Alex, Ten Fabiano, Ten Douglas, Sgt Daniel, Sgt Cabral, Cb Moura e Cb Eduardo.

Por último, gostaria de agradecer a todos os professores e principalmente o meu orientador (Professor Aderlon) que me acompanhou na última trajetória deste percurso.

Como são várias as pessoas que contribuíram, diretamente ou indiretamente, nessa caminhada, agradeço a todos aqueles que fizeram parte da minha formação de cidadão e que eu pudesse, hoje, estar concluindo o curso de Engenharia da Computação.

“Para o bem ou para o mal, o homem é um espírito criativo livre. Isto produz o estranho mundo em que vivemos. Um mundo de criação contínua e, portanto, mudanças e inseguranças contínuas.”

Joyce Cary

Sumário

Índice de Figuras

Índice de Tabelas

Lista de Equações

Glossário de Termos e Abreviaturas

Log – Registro persistido de informações de ação ou evento ocorrido em um sistema computacional.

Pixel – *Picture Element* – Elemento da Figura, considerado como cada ponto de uma imagem.

PSF – *Point Spread Function* – Função de espalhamento de ponto

RGB – *Red, Green, Blue* – Vermelho, Verde, Azul.

SNR – *Signal-to-Noise Ratio* – Relação sinal-ruído.

USB – *Universal Serial Bus* - Tipo de conexão que permite a ligação de periféricos ao computador sem a necessidade de desligá-lo.

SMS - Short message service – Serviço que permite o envio de mensagem de texto entre os telefones celulares.

GPRS - General Packet Radio Service – Serviço que permite o envio e recepção de informações através de uma rede telefônica móvel.

BUS – Conjunto de linhas físicas condutoras que ligam os vários componentes de um computador.

Lista de Símbolos

θ – Valor de limiar para filtros de amaciamento.

λ - Valor de cor de uma imagem

t – Medida de tempo

L – Valor de um nível da cor cinza em uma imagem

r – Raio de uma circunferência

j – Dimensão das linhas de uma matriz de pontos

m – Coordenada do eixo X de uma representação discreta de uma imagem

n – Coordenada do eixo Y de uma representação discreta de uma imagem

O – Número de operações realizadas para execução de um algoritmo

$\mathbf{p_{máx}}$ – Valor máximo de pico em um histograma de brilho

1 INTRODUÇÃO

1.1 Contextualização do Trabalho

Atualmente vivemos em um mundo marcado pela insegurança causada por diversos fatores. Aliado a isso, não são tomadas as devidas medidas pelas autoridades competentes para que estes problemas sejam amenizados.

Por isso, existem diversos sistemas de segurança que tentam garantir a integridade física do indivíduo e também zelar pelas instalações do mesmo.

Neste contexto, a idéia de se ter sistemas capazes de fornecer uma maior segurança é algo necessário. Por isso, sistemas destinados a esta área são cada vez mais promissores e objetos de pesquisas. Sistemas autômatos e controlados computacionalmente estão cada vez mais presentes no mercado e em constante desenvolvimento.

A análise da movimentação humana tornou-se muito importante e aplicável na área de comunicação visual e de realidade virtual. Especialmente, a avaliação da postura humana em tempo real é importante para muitas aplicações como jogos eletrônicos, diversão, segurança e avançados sistemas de interface homem-máquina (TAKAHASHI, 1999).

Atualmente, sistemas de captura de movimento que utilizam dispositivos mecânicos, eletromagnéticos ou acústicos apresentam uma série de problemas, tais como a necessidade do uso de equipamentos especiais e a ocorrência de perda de dados de movimento. O presente trabalho utiliza uma alternativa óptica de captura de movimento, constituída no processo de captura dos dados de câmeras de vídeo.

1.2 Motivação

Infelizmente vivemos em um mundo onde a segurança física das pessoas como também de suas benfeitorias estão comprometidas.

A insegurança enfrentada por muitos cidadãos em suas residências os leva a buscar alternativas para que possam viver com mais tranqüilidade e segurança.

O desconforto em deixar a residência sozinha, mesmo que seja por um pequeno período de tempo, é grande.

Ainda, onde são usados agentes humanos para prover a segurança, temos o problema de confiabilidade nos mesmos. Fator que muitas vezes é de difícil tratamento.

Neste contexto, surgiu a motivação para o desenvolvimento deste projeto. Onde é apresentado um mecanismo que fornece uma maior segurança e tranqüilidade aos cidadãos.

1.3 Objetivo do Projeto

Neste trabalho é desenvolvido um projeto de detecção de movimento para o acionamento do alarme e envio de mensagem para o celular, de forma automática, através do computador. Ainda, é desenvolvido um site onde é possível monitorar e controlar todo o alarme por meio do telefone celular. Neste monitoramento e controle, é possível ver a foto atual do ambiente, ativar e desativar o alarme. Será utilizada uma câmera estática e um hardware. A câmera é responsável pela obtenção da imagem e envio para o computador. O hardware, por sua vez, é responsável pela ligação do mesmo à sirene. Quando da detecção de um intruso (qualquer objeto em movimento) o mecanismo obtém a foto do ambiente monitorado, aciona a sirene e envia uma mensagem ao telefone celular. Após informado do evento, é possível acessar a internet para visualização da foto com o intruso que ocasionou o acionamento do alarme. Ainda é possível controlar o alarme acessando a internet por meio do telefone celular.

São desenvolvidos neste projeto, o Software Controle e Monitoramento, o Software Mobile Web Site Interface e o Hardware de Acionamento da Sirene. A Figura 1.1 mostra um esquemático do funcionamento do sistema.

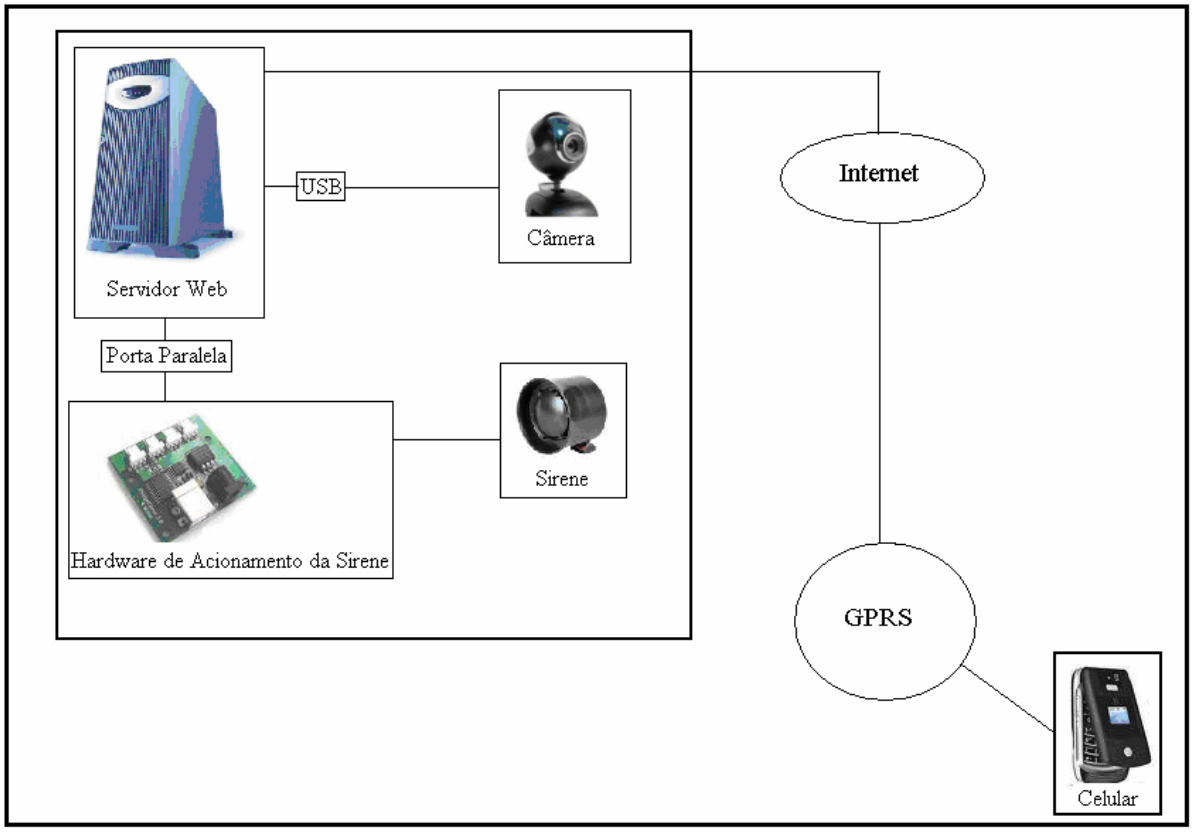


Figura 1-1 – Modelo do Sistema

Fonte: o autor

1.4 Estrutura do Trabalho

Além deste capítulo introdutório, este trabalho está estruturado em cinco capítulos assim distribuídos:

No **Capítulo 2** é apresentada uma introdução teórica sobre análise de imagens, onde são abordados os conceitos essenciais para o entendimento do funcionamento do projeto e, principalmente, do software desenvolvido. Ainda, é apresentado o método de detecção de movimento utilizado e os motivos de sua escolha.

No **Capítulo 3** é detalhada a construção do hardware de acionamento da sirene, descrevendo as etapas e justificativas de montagem.

No **Capítulo 4** é apresentada a construção do Software Controle e Monitoramento e o Software Mobile Web Site Interface. O primeiro é responsável por detectar a presença do possível intruso, mandar uma mensagem para o celular, e ainda ligar e desligar a sirene. Já o segundo, apresenta a interface do usuário e o alarme. São especificadas as tecnologias utilizadas e os motivos de escolha de cada uma. Este capítulo contém ainda a descrição das funcionalidades principais dos módulos da aplicação.

Por fim, no **Capítulo 5** são apresentados os resultados obtidos com a construção deste projeto no que diz respeito à detecção de movimento, o envio de mensagem, o acionamento da sirene e o acesso ao alarme por meio do telefone celular. Ainda, são apresentadas as considerações finais, contendo os testes realizados, as principais conclusões e as dificuldades encontradas.

2 CONCEITOS BÁSICOS SOBRE ANÁLISE DE IMAGENS

Neste capítulo são mostrados os conceitos teóricos sobre análise de imagens que são necessários para a compreensão do projeto desenvolvido. Ainda, é abordado o método de detecção utilizado e o porquê de sua escolha.

2.1 O Sistema de Cores RGB

O sistema de cores RGB (*red, green, blue*) trata-se de um sistema para representar as cores usadas numa exposição de imagem via computador. As cores, vermelho, verde e azul, podem ser combinadas em várias proporções para se obter todas as cores desejadas. Os níveis de R, G, e B podem variar de 0 a 100 por cento de plena intensidade. Cada nível é representado por um número no intervalo de números decimais de 0 a 255 (256 níveis para cada cor), equivalente ao intervalo de números binários de 00000000 a 11111111, ou para hexadecimal 00 a FF. Com isso, o número total de cores disponíveis é 256 x 256 x 256, ou seja, 16.777.216 cores. (WHATIS, 2007). Na Figura 2.1 é ilustrado um mapeamento da combinação das três cores do sistema RGB (vermelho, verde e azul).

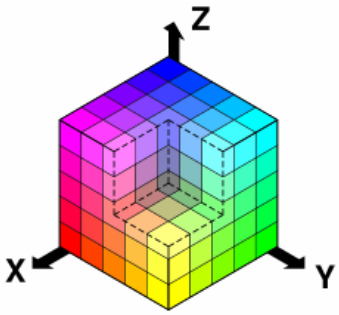


Figura 2-1 - Modelo RGB mapeado para um cubo em corte
Fonte: WHATIS, 2007.

2.2 Escala de Cinza

As imagens exibidas em escala de cinza ou *grayscale*, são imagens em que o valor de cada pixel é uma única amostra. Isso quer dizer que no sistema RGB (*red, green e blue*) a intensidade de cada uma das três cores é a mesma.

Sendo assim, as imagens geradas são formadas por máscaras de cinza que variam do preto (menor intensidade) ao branco (maior intensidade).

Ainda, as imagens neste formato são diferentes das imagens em preto-e-branco, pois estas, por sua vez, são imagens com somente duas cores, o preto e o branco. Já as imagens em *grayscale* possuem muitas variações de cores entre o preto e o branco.

2.3 Análise de Imagens

Com o surgimento da tecnologia digital moderna tornou-se possível a utilização de sinais multidimensionais através de sistemas. Estes variam desde simples circuitos digitais a avançados computadores paralelos. Um dos objetivos dessa utilização destaca-se a análise de imagem (YOUNG, 2004):

Esta seção abordou os conceitos fundamentais da análise de imagem. Embora restritos a imagens bidimensionais (2D), a maioria dos conceitos e técnicas que serão descritas possam ser adaptadas para três ou mais dimensões sem prejuízo de conceito (YOUNG, 2004).

2.3.1 Definições Básicas

2.3.1.1 Imagem

A formação de uma imagem pode ser definida como uma função de duas variáveis reais *a(x,y).* Onde *a* é a amplitude (de brilho, por exemplo) da imagem nas coordenadas *x,y*. Ainda, uma imagem pode conter sub-imagens que normalmente são chamadas de *região de interesse*, ou simplesmente *região*. Este conceito mostra o fato de que imagens normalmente contêm conjuntos de objetos os quais podem ser individualmente a base para uma região. (YOUNG, 2004).

A amplitude (*a*) de uma imagem é normalmente representada por números reais ou inteiros. Embora em casos como representação por imagem de ressonância magnética, a medida física direta forneça um número complexo na forma de uma magnitude real e uma fase real (YOUNG, 2004). Neste trabalho, apenas são apresentados casos de amplitudes reais ou inteiras.

2.3.1.2 Imagem Digital

A formação de uma imagem digital *a[m,n]* descrita em um espaço discreto 2D é derivada de uma imagem analógica *a(x,y)* em espaço contínuo 2D, através de um processo matemático de amostragem, quantificação e ou codificação comumente chamado de *digitalização* (YOUNG, 2004).

2.3.1.3 Pixel

A imagem contínua em 2D *a(x,y)* é dividida em *N* linhas e *M* colunas. O pixel é a intersecção de uma linha e uma coluna. O valor atribuído às coordenadas inteiras *[m,n]* com *{m=0,1,2,...,M-1} e {n=0,1,2,...,n-1}* é representado pela função *a[m,n]*. De fato, na maioria dos casos, *a(x,y)* - que deve ser considerada como o sinal físico na face de um sensor 2D – é uma função de muitas variáveis, como profundidade *(z),* cor *(λ)* e tempo(*t)*. Em todo este capítulo são consideradas apenas imagens 2D, monocromáticas e estáticas (YOUNG, 2004). Na Figura 2.2 é ilustrado o processo de digitalização de uma imagem contínua.

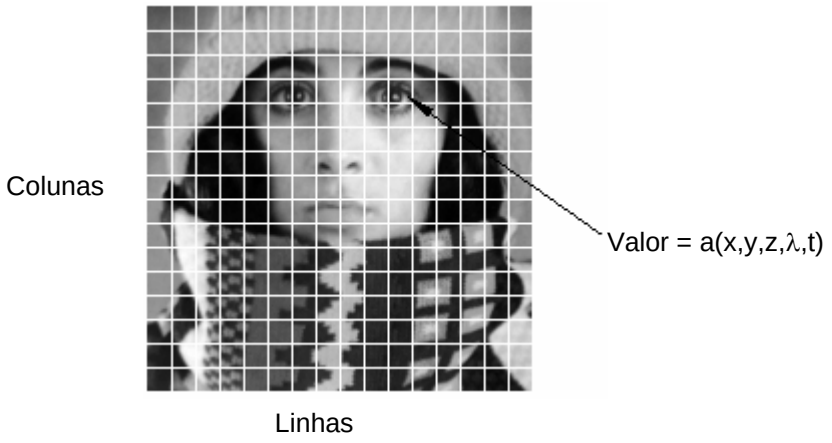


Figura 2-2 - Digitalização de uma imagem contínua.
Fonte: YOUNG, 2004.

Na Figura 2.2 é apresentada uma imagem dividida em 16 linhas e 16 colunas. O valor atribuído a cada pixel é a média do valor de brilho do pixel, arredondado para o inteiro mais próximo. O processo de se representar a amplitude de um sinal 2D em uma dada coordenada através de um inteiro, que no caso mostra um valor de *L* diferentes níveis de cinza, é usualmente chamado de *quantização de amplitude*, ou simplesmente *quantização* (YOUNG, 2004).

2.3.2 Algoritmos

Normalmente, as principais operações são normalmente divididas em quatro tipos (YOUNG, 2004):

a) Operações baseadas no histograma da imagem;
b) Operações baseadas em matemática simples;
c) Operações baseadas em convolução e;
d) Operações baseadas em morfologia matemática.

Estas quatro operações podem ainda ser classificadas quanto à natureza de sua implementação como sendo: operações aplicadas ponto-a-ponto, operações locais, ou operações globais (YOUNG, 2004).

Porém, aqui é mostrada apenas a operação matemática de diferença que faz parte integrante do contexto deste trabalho.

2.3.2.1 Operações Matemáticas

A aritmética binária (Booleanos) apresenta as operações matemáticas que constituem a base de uma série de ferramentas para análise de imagens. As operações detalhadas a seguir (Equações 2.1, 2.2, 2.3, 2.4 e 2.5) são operações aplicadas para cada ponto da imagem (pixel a pixel) e permitem um variado conjunto de eficientes implementações (YOUNG, 2004).

$$OR \quad c = a + b \tag{2.1}$$

$$NOT \quad c = \overline{a} \tag{2.2}$$

$$XOR \quad c = a \oplus b = a \bullet \overline{b} + \overline{a} \bullet b \tag{2.3}$$

$$AND \quad c = a \bullet b \tag{2.4}$$

$$SUB \quad c = a \setminus b = a - b = a \bullet \overline{b} \tag{2.5}$$

As Tabelas 2.1a, 2.1b, 2.1c, 2.1d e 2.1e contêm as definições das operações binárias aplicadas ponto a ponto.

Tabela 2.1 - Definição das operações binárias

NOT a	
0	1
1	0

↑ input ↑ output

(a)

OR a \ b	0	1
0	0	1
1	1	1

(b)

AND a \ b	0	1
0	0	0
1	0	1

(c)

XOR a \ b	0	1
0	0	1
1	1	0

(d)

SUB a \ b	0	1
0	0	0
1	1	0

(e)

Na Figura 2.3 é demonstrada a aplicação das operações binárias aplicadas ponto a ponto nas imagens.

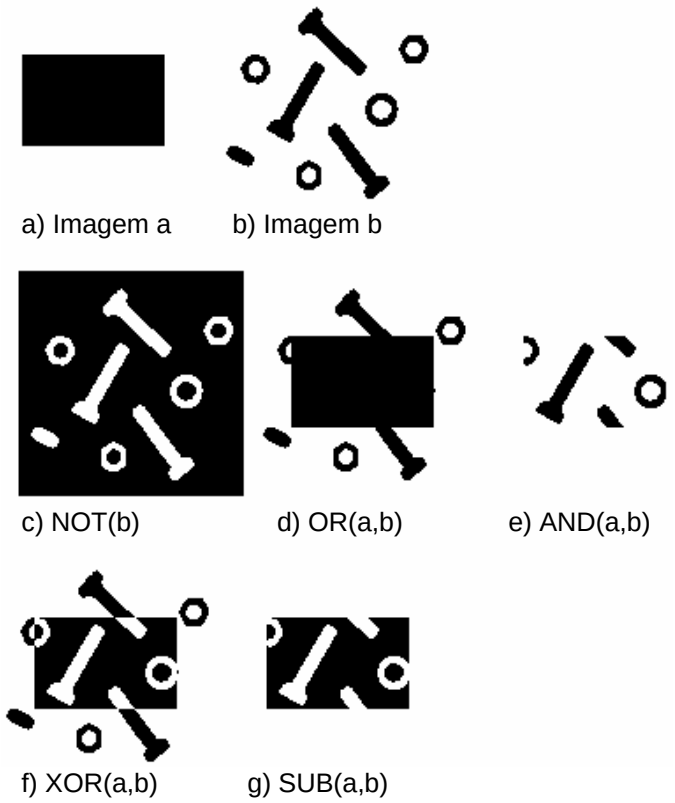


Figura 2-3 - Exemplo de operações binárias em pontos
Fonte: YOUNG, 2004.

2.3.2.1.1 Diferença

O filtro de diferença é formado pela combinação das operações binárias simples NOT e AND, da forma *NOT(AND(a,b))*. Nesta comparação de duas imagens ponto-a-ponto é retornado como imagem de saída apenas os pixels que não coincidem em *a* e *b* (YOUNG, 2004). Na Figura 2.4 é mostrado o resultado de um filtro de diferença entre duas imagens a e b.

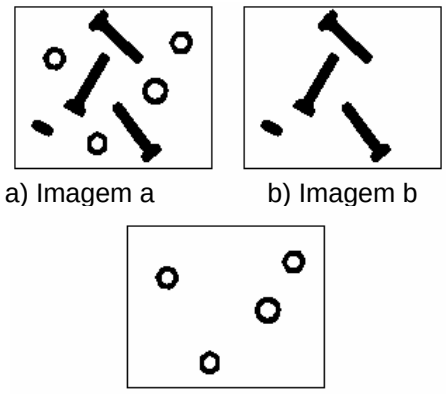


Figura 2-4 - Exemplo de filtro de Diferença
Fonte: YOUNG, 2004.

Percebe-se na Figura 2.4 que a operação compara ponto-a-ponto as imagens *a* e *b* retornando para cada ponto o resultado da operação *NOT(AND(a,b))*, gerando como imagem de saída apenas os pixels não coincidentes nas imagens de entrada.

2.3.3 Técnicas de Segmentação de Imagens

Anteriormente, foram apresentados os algoritmos que podem ser combinados para resolver determinados problemas em análise de imagem, Por exemplo: *correção de sombra*, *realce básico, técnicas de restauração* e *segmentação.* (YOUNG, 2004).

Como no desenvolvimento do projeto foi utilizada a técnica de segmentação *limiar* para a detecção de movimento, a mesma será detalhada nesta seção.

Com relação às técnicas segmentação é importante destacar ainda que (YOUNG, 2004):

a) Não há uma técnica de segmentação universalmente aplicável que funcione para todas as imagens, e;
b) Nenhuma técnica de segmentação é perfeita.

2.3.3.1 Limiar (*Thresholding*)

Esta técnica de segmentação foi a utilizada para a detecção do movimento e baseia-se no seguinte conceito. Um parâmetro θ chamado de *limiar de brilho* é escolhido e aplicado à imagem *a[m,n]* segundo a seguinte condição:

Se $a[m,n] \geq \theta$ *então* *a[m,n] = objeto = 1*
Senão *a[m,n] = fundo = 0*

Esta modalidade do algoritmo supõe que os objetos de interesse estão iluminados sob um fundo escuro. Para objetos escuros num fundo iluminado usa-se:

Se $a[m,n] < \theta$ *então* *a[m,n] = objeto = 1*
Senão *a[m,n] = fundo = 0*

Como abordado nesta seção, vimos que escolher o valor apropriado do *limiar* é a questão principal nos processos de segmentação. Porém, como não existe um método universal para a seleção de um *limiar* com funcionamento garantido com qualquer imagem, são abordados, na seqüência, as mais normalmente utilizadas (YOUNG, 2004).

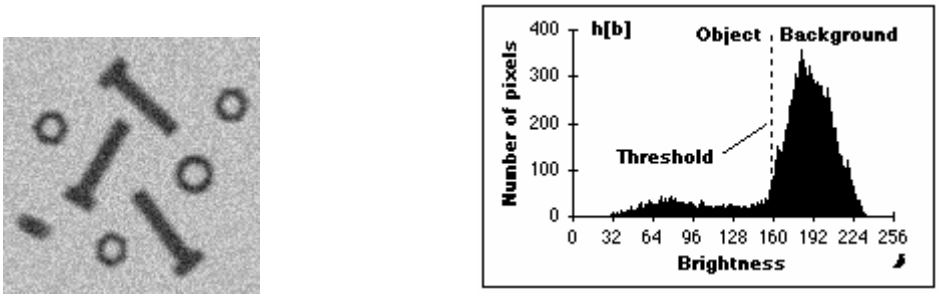
2.3.3.1.1 Limiar Fixo

Uma alternativa para escolha de um θ apropriado é utilizar um limiar que seja escolhido independentemente dos dados de imagem. Se for sabido que se lida com imagens de alto-contraste, onde os objetos são muito escuros e o fundo é homogêneo e muito iluminado, então um limiar constante de 128 numa escala de 0 a

255 talvez seja suficientemente exato. Por exatidão entende-se que o número de pixels falsamente classificados é mantido ao mínimo (YOUNG, 2004).

2.3.3.1.2 Limiares Derivados de Histogramas

Na maioria de casos, o limiar é escolhido a partir do histograma de brilho da região ou imagem que se deseja dividir. Uma imagem e seu histograma de brilho associado são mostrados na Figura 2.5. Os pixels sob o limiar ($a[m,n] < \theta$) serão marcados como pixels de objeto e os acima do limiar estarão marcados como pixels de fundo (YOUNG, 2004).



(a) Imagem a ser processada (b) Histograma de brilho da imagem

Figura 2-5 - Imagem de entrada e histograma de brilho
Fonte: YOUNG, 2004.

2.3.3.1.3 Algoritmo Isodata

O histograma é inicialmente dividido em duas partes, usando um valor de limiar inicial correspondente a metade do alcance dinâmico máximo. A amostra central dos valores de cinza associados aos pixels de primeiro plano e a amostra central dos valores de cinza associados aos pixels de fundo é calculada. Um novo valor de limiar é calculado como sendo a média destas duas amostras. Este

processo é repetido recursivamente, baseado no novo limiar, até que o valor de limiar não mude mais (YOUNG 2004).

2.3.3.1.4 Algoritmo de Simetria de Fundo

Esta técnica supõe um pico dominante distinto para o fundo, que é simétrico ao seu máximo. O pico máximo ($p_{máx}$) corresponde ao valor máximo do histograma. O algoritmo então procura no lado dos pixels não pertencentes a objetos do máximo para determinar um ponto p% (YOUNG, 2004).

Utilizando o exemplo da Figura 2.5b, onde os pixels de objeto são localizados à esquerda do pico de fundo (brilho = 183), significaria localizar a direita desse pico, por exemplo, o valor de 95%. Neste valor de brilho, 5% dos pixels estão à direita (acima) desse valor. Isto ocorre para o valor de brilho 216 na Figura 2.5b. Em virtude da suposta simetria, é utilizado como limiar um deslocamento à esquerda do máximo, que é igual ao deslocamento à direita onde o p% foi encontrado. Para o exemplo em questão, este valor de limiar seria dado por 183 - (216 - 183) = 150. Tal valor é obtido de acordo com a Equação 2.6 (YOUNG, 2004).

$$\theta = P_{máx} - (P\% - P_{máx}) \qquad (2.6)$$

2.3.3.1.5 Algoritmo do Triângulo

Uma linha é traçada entre o máximo do histograma de brilho B_{max} e o menor valor B_{min}= *(P=0)%* na imagem. A distância *d* entre a linha e o histograma *h[B]* é calculada para todos os valores de *B*, onde $B = B_{min}$ até $B = B_{max}$. O valor de brilho B_0, onde a distância entre $h[B_0]$ e a linha é máxima, corresponde ao valor de limiar, ou seja, $\theta = B_0$. Esta técnica é particularmente eficiente quando os pixels de objeto

produzem um pico fraco no histograma. A Figura 2.13 ilustra a obtenção da distância *d* a partir do histograma de brilho da imagem, utilizada no algoritmo do triângulo (YOUNG, 2004).

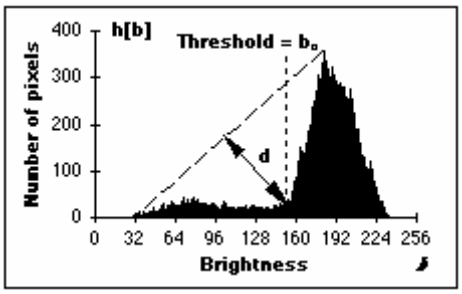


Figura 2-6 - Ilustração do algoritmo do triângulo
Fonte: YOUNG, 2004.

Os três procedimentos descritos anteriormente resultam nos valores θ=139 para o algoritmo de Isodata, e θ=150 para o algoritmo de simetria de fundo com 5% de nível, e θ=152 para o algoritmo de triângulo, considerando a imagem da Figura 2.5a (YOUNG, 2004).

O filtro de limiar (*thresholding*) não precisa necessariamente ser aplicado a imagens inteiras, podendo ser utilizado em regiões específicas da imagem. Chow e Kaneko (YOUNG, 2004) desenvolveram um algoritmo em que a imagem de *NxM* é dividida em regiões não sobrepostas. Para cada região um limiar é calculado e os valores resultantes de limiar são montados (interpolados) formando uma superfície de limiarização para a imagem inteira. As regiões devem ser de tamanho razoável, de modo que haja um número suficiente de pixels em cada região para se fazer uma estimativa do histograma e, conseqüentemente, do valor de limiar. A utilidade deste procedimento - como tantos outros - depende da aplicação em questão (YOUNG, 2004).

2.4 Detecção de Movimento por Análise de Imagens

2.4.1 Preâmbulo

Em ambientes onde se queira detectar pessoas ou objetos, geralmente não há controle sobre o que haverá no plano de fundo da imagem capturada (e como conseqüência, não se tem muito controle sobre sua complexidade), bem como na intensidade de iluminação e outros objetos em movimento. As pessoas também têm sua aparência sujeita a uma enorme variação, desde vestimentas, cor da pele, ausência de algum dos membros, altura e diferentes possibilidades de movimentos (SOARES, 2004).

Então, todos estes fatores dificultam a identificação de formas na imagem, embora algumas tentativas já tenham sido realizadas com taxas de sucesso razoáveis (SOARES, 2004).

2.4.2 Método: Diferença entre dois quadros sucessivos

O movimento é um poderoso artifício empregado por seres humanos e animais para extrair objetos de interesse de um cenário de fundo (GONZALEZ[1], 1992). Neste método, calcula-se para cada quadro a diferença entre este e o anterior, o qual é tomado como referência. Assim, se uma imagem for expressa como $I=f(x,y,t)$, onde I é um valor representando a intensidade luminosa no ponto de coordenadas (x,y) no instante t, tem-se que (SOARES, 2004):

$$\Delta I=|f(x,y,t_2)- f(x,y,t1)| \qquad (2.7)$$

[1] GONZALEZ R.C., Woods R. E. Digital Image Processing - Addison-Wesley Publishing Company, 1992, ISBN 0-201-50803-6, p. 632

onde ΔI é o módulo da diferença entre a imagem no instante t_2 (imagem atual) e a imagem no instante t_1 (imagem anterior).

Em um ambiente controlado, com pouca variação, a diferença entre os dois quadros sucessivos terá valores iguais ou muito próximos a zero nas regiões onde existe pouca ou nenhuma variação na intensidade luminosa, e valores diferentes de zero nas regiões onde objetos estão em movimento. Pode-se considerar que variações, dentro de certa faixa de tamanho (extensão no plano) e proporções, correspondam a uma pessoa em movimento e, fora desta faixa, simples objetos (SOARES, 2004).

Um parâmetro a ser ajustado no sistema é o valor de limiarização da diferença entre as duas imagens. Este limiar indicará o que é realmente uma diferença entre as duas imagens (considerado como uma variação grande) e o que pode ser encarado como ruído (variação pequena). Valores muito baixos de limiar produzirão diferenças por toda imagem, uma vez que duas imagens sucessivas não são perfeitamente iguais devido a erros introduzidos no processo de aquisição de imagens e variações na iluminação. Valores de pixel inferiores ao limiar são substituídos por zero e valores acima por 255. Os valores iguais a 255 são identificados como presença de movimento (SOARES, 2004).

Valores elevados de limiar não permitirão a detecção de diferenças entre as duas imagens. Valores de limiar entre os extremos produzem nas regiões onde objetos ou pessoas encontram-se em movimento, um contorno da silhueta da pessoa, correspondente ao movimento que ocorreu durante o tempo de transição de um quadro (SOARES, 2004).

3 DESCRIÇÃO DO *HARDWARE*

Este capítulo trata da descrição do hardware construído e dos componentes eletrônicos utilizados. Também será descrita as características da câmera utilizada.

3.1 Porta Paralela

A porta paralela é um dos meios de comunicação usada entre o computador e os demais periféricos existentes atualmente. Neste projeto, ela é a interface de comunicação usada entre o hardware de acionamento da sirene e o computador.

3.1.1 Tipos de Transmissão

A comunicação via porta paralela apresenta dois meios de transmissão que merecem atenção. Os mesmo são descritos a seguir.

3.1.1.1 Unidirecional (SPP-Standard Parallel Port)

Neste tipo de transmissão, apresenta uma taxa de transferência que pode chegar a 150000 B/s e se comunica com a CPU com um BUS de dados de 8 bits. Porém, para a transmissão entre periféricos são utilizados 4 bits por vez (Rogercom)[2].

[2] Rogercom **[Home Page].** 2007. Disponível em: <http://www.rogercom.com>

3.1.1.2 Bidirecional (EPP – Enhanced Parallel Port)

Já na transmissão bidirecional, pode atingir uma taxa de transferência de 2 MB/s e se comunica com a CPU com um BUS de dados de 32 bits. Ainda, na transferência entre periféricos são utilizados 8 bits por vez. Isso é possível pois neste tipo de transmissão é utilizado acesso direto à memória sem a necessidade do uso do processador. Também, é utilizado um buffer de 16 bytes (Rogercom).

3.1.2 Endereçamento

Existem dois tipos de endereços utilizados para a comunicação da porta paralela. São eles: endereçamento de Porta(378h) e de Memória(0000:0408). Neste projeto o software de controle e monitoramento acessa a porta paralela pelo endereço de porta.

3.1.3 Registradores

A porta paralela possui os registradores de dados, status e controle. Sua descrição e o respectivo endereço são mostrados na figura abaixo.

Nome	Endereços LPT1	Descrição
Registro de Dados	378h	Envia um byte
Registro de Status	379h	Ler o Status
Registro de Controle	37Ah	Envia dados de controle

Figura 3-1 Registradores

Fonte: Rogercom.

3.1.4 Conector DB25

Neste conector é ligado o cabo paralelo para o envio e leitura de dados. Ele possui 25 pinos como mostrado na figura 3-2.

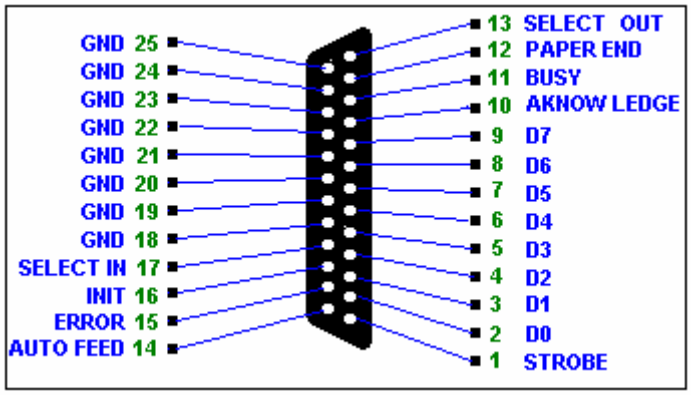


Figura 3-2 - Conector DB25

Fonte: Rogercom.

Quando a tensão em qualquer um desses pinos está entre 3,1V e 5V, indica que os mesmos estão em nível lógico 1 e quando a mesma está entre 0V e 0,4V, indica nível lógico 0 (Rogercom).

Na figura 3-3 é apresentado o funcionamento do DB25 no modo SPP(Standard Parallel Port):

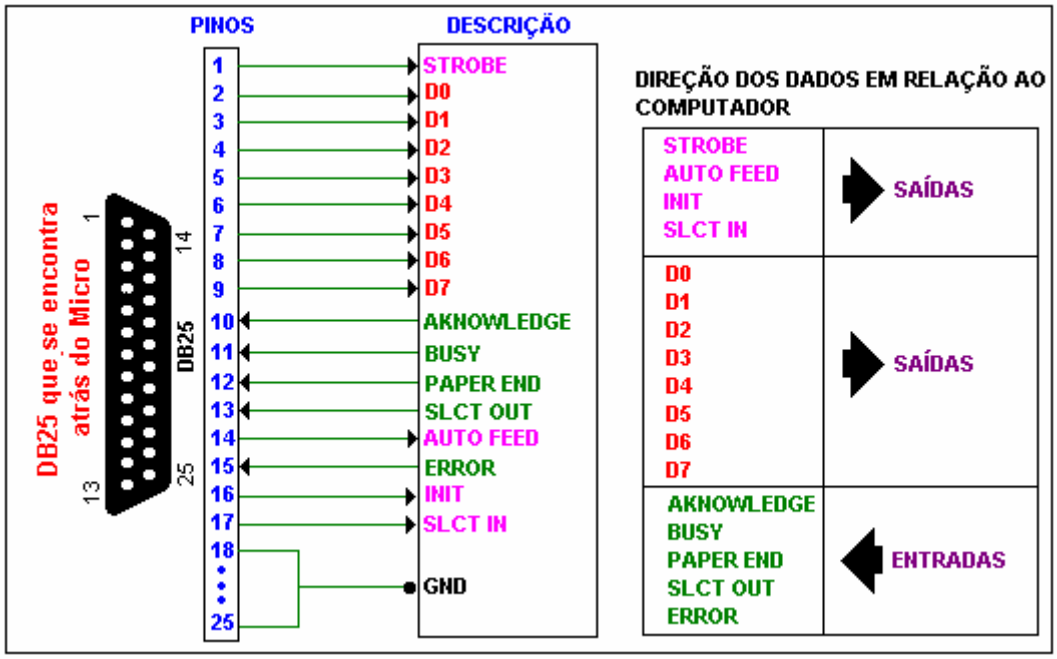


Figura 3-3 – Quadro de funcionamento da porta paralela em modo padrão (SPP)
Fonte: Rogercom.

3.2 Descrição do Hardware de Acionamento da Sirene

Nesta seção são listados os componentes utilizados para confecção do hardware e uma breve descrição de funcionamento do mesmo.

3.2.1 Componentes Utilizados

Os componentes utilizados para a confecção do Hardware de Acionamento da Sirene foram:

1) 01 (um) diodo 1N4148;
2) 01 (um) resistor (R1) de 2,2K ohm;
3) 01 (um) resistor (R2) de 470 ohm;
4) 01 (um) transistor BD 137;
5) 01 (um) circuito integrado buffer 74LS541;

6) 01 (um) relê com 12 volts na bobina; e

7) 01 (uma) sirene com alimentação de 12 volts.

3.2.2 Funcionamento

A interação do software controle e monitoramento com o hardware de acionamento da sirene ocorre por meio da porta paralela. Para isso, é utilizada a DLL *Inpout32*.

Neste projeto foi utilizado o pino 2 da porta paralela para ligar e desligar a sirene. A sirene é acionada quando o pino 2 está em nível lógico 1. Para que isso ocorra, o Software Controle e Monitoramento por meio da DLL *Inpout32* escreve 00000001 no endereço 378h. Desta forma, o transistor BD137, que funciona como uma chave, fecha o circuito do relê o acionando. E este, por sua vez, liga a sirene.

Na figura 3-4 está apresentado o modelo do circuito do hardware implementado neste projeto.

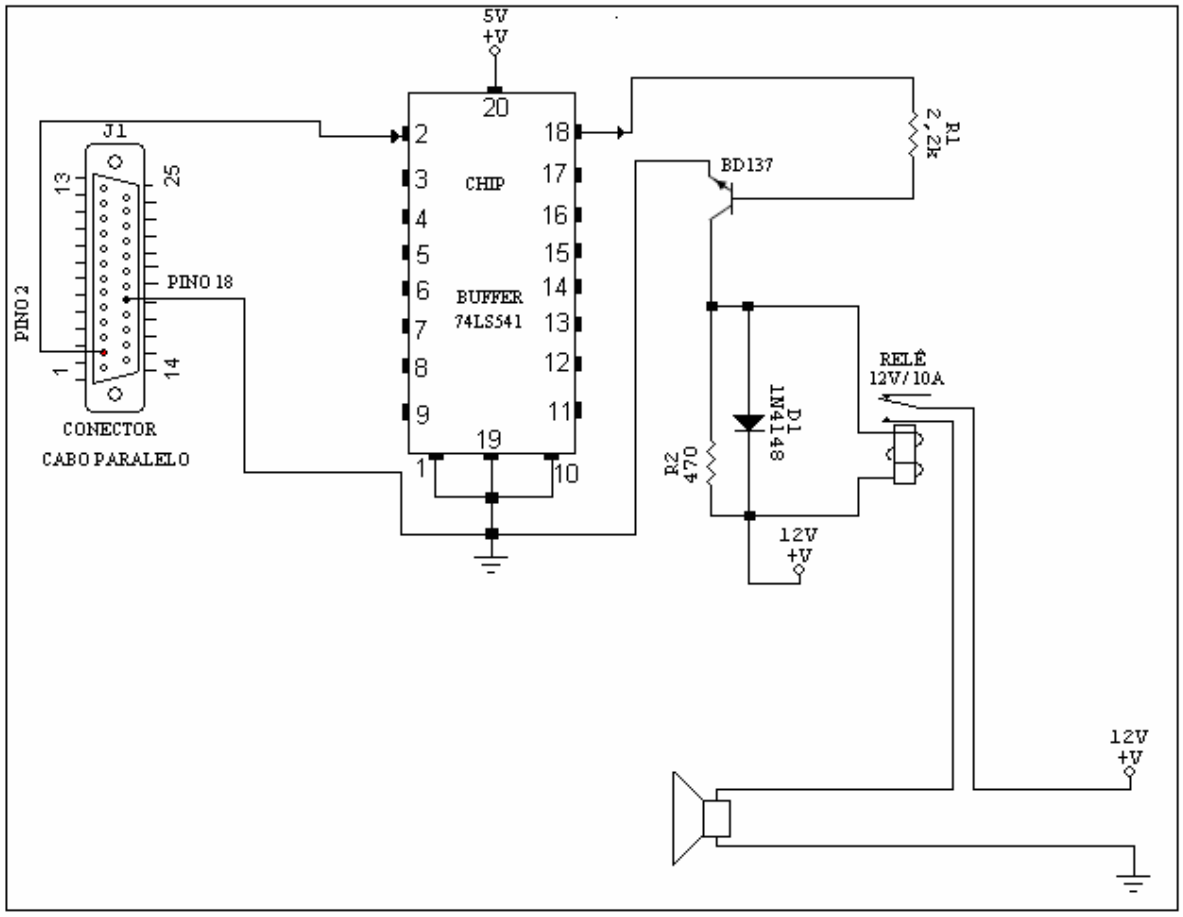


Figura 3-4 Modelo do circuito do hardware

Fonte: Rogercom

Modificado pelo autor

3.3 Câmera

A câmera utilizada é uma câmera web comum, marca A4Tech, modelo PK-635M, colorida, com definição máxima de 640 X 480 e 350000 pixels. Ela possui recurso de balanço dos brancos automático, o que reduz o nível de ruído nas imagens por variação de intensidade luminosa do ambiente.

A conexão com o computador é feita através de uma porta USB, sendo ativada automaticamente através do software controle e monitoramento, o qual captura os quadros (*frames*) para a detecção do movimento.

A câmera é posicionada no ambiente onde é realizado o controle e monitoramento do mesmo.

4 DESCRIÇÃO DO *SOFTWARE*

Este capítulo detalha a construção dos *softwares* Controle e Monitoramento e do Mobile Web Site Interface. O primeiro está dividido em três módulos principais: detecção de movimento, acionamento da sirente e envio de mensagem (SMS). Já o segundo, se refere ao site que é acessado pelo telefone celular para o controle e monitoramento do ambiente desejado. Há ainda um tópico específico para apresentação do framework de código aberto Aforge.Net, utilizado no desenvolvimento e que agrega uma série de funcionalidades genéricas para a detectão do movimento, tais como os filtros e algoritmos descritos no capítulo 2. Também é retratado a biblioteca utilizada para o envio da mensagem (SMS).

4.1 Tecnologia Utilizada

Para a construção do software de controle e monitoramento foi utilizado o Microsoft Framework 2.0 [3] e linguagem C#. Já para o mobile web site interface, além destas tecnologias, foi utilizado a linguagem ASP.Net. A escolha deu-se pela familiaridade com o ambiente e pela facilidade que a linguagem traz para implementação com orientação a objeto.

4.2 AForge.Net *Framework*

O Aforge.Net é um *framework* de código aberto (*open source*), desenvolvido por Andrew Kirillov[4], licenciado sob os termos da *GNU General Public License 2.0*, escrito em linguagem C#, com o objetivo de auxiliar desenvolvedores e pesquisadores nos campos da “Visão por Computador” e “Inteligência Artificial”,

[3] Ver http://msdn2.microsoft.com/pt-br/netframework/default.aspx

[4] O código fonte, bem como detalhes do projeto, podem ser encontrados em http://code.google.com/p/aforge/

como processamento de imagens, redes neurais, algoritmos de genética e aprendizagem de máquinas.

Ele é composto de 5 bibliotecas principais, além de algumas de uso genérico (apoio). As principais são:

a) AForge.Imaging – Biblioteca para rotinas de processamento de imagens e filtros;
b) AForge.Neuro – Biblioteca para computação em redes neurais;
c) AForge.Genetic – Biblioteca para algoritmos na área de genética;
d) AForge.Vision – Biblioteca para Visão por Computador;
e) AForge.Machine Learning – Biblioteca para aprendizagem de máquinas.

Porém, neste projeto foi utilizada apenas a biblioteca AForge.Imaging como parte integrante do software controle e monitoramento para a detecção de movimento.

4.3 Software Controle e Monitoramento

Este software é responsável pela detecção do movimento, envio da mensagem para o celular e acionamento da sirene. Suas funções são detalhadas a seguir.

4.3.1 Módulo de Detecção de Movimento

O módulo de detecção de movimento é o módulo mais importante deste projeto. Isso deve ao fato de que ele nos traz uma excelente alternativa perante os outros meios de detecção existentes. Ele implementa as seguintes funcionalidades principais, que serão descritas em detalhes:

a) Aquisição de imagens;
b) Pré-processamento das imagens e;
c) Aplicação dos filtros e algoritmos para detecção de movimento.

4.3.1.1 Aquisição de Imagens

Foram utilizadas classes que implementam as funções para conexão com a câmera, captura do *streaming* de vídeo e sua conversão para o formato “mapa de bits”. As subseções 4.3.1.1.1, 4.3.1.1.2, 4.3.1.1.3 as descrevem.

4.3.1.1.1 Interface IVideoSource

Esta classe é uma interface para qualquer objeto que forneça um *streaming* de vídeo, provendo compatibilidade entre diversas fontes de vídeo como, por exemplo, arquivos em formato AVI, JPEG streamings e MJPEG streamings. Além de declarar propriedades comuns, ela possui um *Event Handler* para tratar os frames fornecidos pela fonte produtora de vídeo.

4.3.1.1.2 Classe CaptureDevice

Esta classe implementa a interface abstrata *IVideoSource* para o caso de um dispositivo local (conectado diretamente ao computador) ser utilizado como fonte de streaming de vídeo.

4.3.1.1.3 Classe Camera

Esta classe implementa um objeto como função principal publicar os quadros (*frames*) obtidos, encapsulando o objeto *videosource* e provendo serviços de alto nível para o controle da câmera. É também responsável por manter a instância do tipo de detector de movimento utilizado.

4.3.1.2 Pré-processamento de Imagens

Esta funcionalidade foi embutida diretamente nas classes que implementam a interface *IMotionDetector*. São basicamente conversão para escala de cinza e simplificação dos valores de brilho dos pixels da imagem para números inteiros (média dos canais).

4.3.1.3 Aplicação dos Filtros e Algoritmos para Detecção de Movimento

Neste ponto ocorre a utilização dos filtros providos pelo *framework* AForge.Net, cuja teoria foi introduzida no capítulo 2. O primeiro passo consiste em aplicar o filtro de diferença entre as imagens do frame anterior (*overlay image*) e a imagem do atual, conforme descrito na subseção 2.3.2.1.1.

A imagem resultante é então limiarizada através de um filtro baseado no histograma de brilho da imagem, conforme descrito na subseção 2.3.3.1.2.

Como descrito na subseção 2.4.2, o valor do limiar deve ser devidamente ajustado para que meros ruídos não sejam interpretados como movimento. Isso é feito na classe *Camera* ajustando o valor da variável *alarmLevel.*

4.3.2 Acionamento da Sirene

Para o acionamento da sirene foi utilizada a DLL *Inpout32* junto ao software controle e monitoramento. Esta DLL permite a escrita e a leitura na porta paralela. Com isso, é possível ligar e desligar a sirene escrevendo nível lógico 1 ou 0 na mesma.

4.3.3 Envio de Mensagem (SMS)

O envio de mensagem é realizado com a utilização da API *SMS Web Service SOAP API* no software de controle e monitoramento. Esta API é desenvolvida em C# pela empresa *Connection Software* e disponibilizada gratuitamente. Porém, para que seja possível o envio da mensagem, é necessário adquirir créditos junto a mesma. O interessante é que os preços das mensagens variam de acordo com a garantia de serviço oferecido [Connection Software].

O tempo gasto para o recebimento da mensagem pelo celular após o disparo do alarme foi satisfatório.

4.4 Software Mobile Web Site Interface

Este software é o site que é acessado através do telefone celular para o controle e monitoramento de todo o sistema de alarme.

4.4.1 Funcionamento

Este software funciona da seguinte maneira. O usuário informa o nome e senha na página de login.

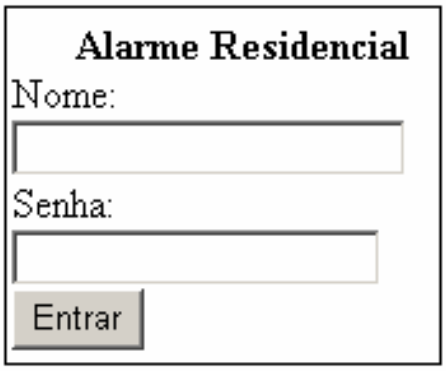


Figura 4-1 – Página de login

Fonte: o autor.

Após a devida validação, ele tem acesso à página de controle e monitoramento. Nesta, pode ser verificado o status do sistema e, ainda, é possível acionar ou desativar o mesmo.

Figura 4-2 Página de controle e monitoramento

Nesta página, caso o status do sistema seja *ativado,* a foto disponibilizada será a foto obtida no último minuto pelo software de controle e monitoramento. Agora, se o status do sistema for disparado, a foto disponibilizada será a foto obtida do intruso dois segundos após o momento da detecção do mesmo.

4.5 Comunicação Entre os Softwares Controle e Monitoramento e o Mobile Web Site Interface

Para a comunicação entre os dois softwares foi utilizado o SGBD (Sistema de Gerenciamento de Banco de Dados) MySql Server. Inicialmente foi utilizado um arquivo XML. Porém, o seu uso não apresentou bons resultados tendo em vista que os dois softwares tentavam utilizá-lo ao mesmo tempo. Sendo assim, o uso de um SGBD foi necessário.

Nesta seção é mostrada a tecnologia do MySql Server e retratado a comunicação entre os dois softwares por meio do banco de dados *bd_alarme*.

4.5.1 SGBD MySql Server

O SGBD MySql Server é desenvolvido pela empresa MySQL AB. Ela o disponibiliza a custo zero sob a licença GPL (GNU General Public License), ou sob uma licença comercial para quem pretende não seguir os termos da GPL [MySql].

A MySQL AB surgiu na Suécia e foi criada por David Axmark, Allan Larsson e Michael “Monty” Widenius [MySql].

Atualmente, o MySQL é o SGBD de código aberto mais popular do mundo com mais de 6 milhões de instalações [MySql].

Dentre as razões do sucesso do MySQL destaca-se a consistência, alta performance e confiabilidade. Além de tudo isso, destaca-se também por ser de fácil

uso. Ele funciona em mais de 20 plataformas dentre as quais se destacam o Linux, Windows, HP-UX, AIX e Netware [MySql].

Porém, apesar da grandeza do MySql, neste projeto ele foi utilizado apenas como um meio de comunicação entre os dois softwares desenvolvidos como é descrito a seguir.

4.5.2 O Banco de Dados *bd_alarme*

O banco de dados *bd_alarme* foi desenvolvido para a comunicação entre os softwares. Ainda, ele possui a tabela *usuario* onde são cadastrados todos os usuários que tenham permissão de controle e monitoramento do sistema. A figura 4-3 mostra o modelo do banco *bd_alarme.*

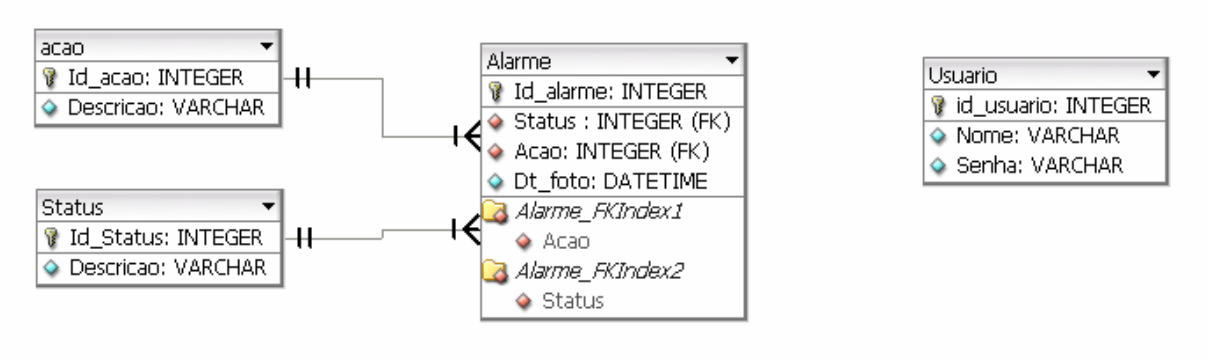


Figura 4-3 - Modelo do banco de dados *bd_alarme*

4.5.3 Funcionamento da Comunicação

A comunicação ocorre basicamente por meio dos campos *Status* e *Acao* que estão na tabela *Alarme.*

O software de controle e monitoramento faz a leitura, a todo o momento, desses campos. Quando os mesmos forem diferentes indica que ocorreu uma ação

e o mesmo a executa. Caso contrário, ele não realiza nenhuma ação. Esta ação é solicitada quando a câmera verifica um movimento (disparar) ou quando o usuário acessa o site (software mobile web site interface) e solicita uma ação (ativar, desativar).

4.5.4 Permissão de Controle e Monitoramento dos Usuários

Ainda, é no banco *bd_alarme* que se encontram as permissões de controle e monitoramento dos usuários. Ele contém a tabela *usuario* onde são cadastrados aqueles que podem acessar o sistema.

Esta tabela apresenta os campos de *Nome* e *Senha* que garantem a segurança de acesso. Então, para que o usuário consiga acessar o site (software mobile web site interface) é necessário que ele tenha um registro com o seu nome e sua devida senha.

5 RESULTADOS OBTIDOS

Neste capítulo são retratados os resultados obtidos na execução desse projeto. Os mesmos são divididos em quatro partes: detecção de movimento, envio de mensagem, acionamento da sirene e acesso ao alarme por meio do telefone celular. Ainda são retratadas as considerações finais com os testes realizados, as dificuldades encontradas e as conclusões.

5.1 Detecção de Movimento

O uso da biblioteca AForge.Imaging para a detecção de movimento foi de grande importância. Isso deve-se ao fato da mesma implementar filtros e algoritmos como o método da diferença entre dois quadros sucessivos e a limiarização os quais, por sua vez, são fundamentais para a detecção do movimento como foi descrito neste trabalho.

Deste modo, os testes realizados mostraram satisfatórios como o esperado.

5.2 Envio de Mensagem

O software controle e monitoramento utiliza uma biblioteca para o envio da mensagem. Esta biblioteca é disponibilizada gratuitamente pela empresa *Connection Software*. E esta empresa, por sua vez, faz a garantia do envio da mensagem para o telefone celular. Esta garantia mostrou satisfatória tendo em vista que a mensagem chega ao celular em torno de vinte segundos após o disparo do alarme.

5.3 Acionamento da Sirene

Aqui, temos dois itens que merecem destaques: o hardware de acionamento da sirene e a DLL *Inpout32.dll.*

O hardware de acionamento da sirene mostrou-se eficaz e atendeu ao seu objetivo.

A DLL *Inpout32.dll* inserida no software de controle e monitoramento para a escrita na porta paralela é de fácil utilização e escreve nível lógico 1 e 0 na mesma. Assim, ligando e desligando a sirene respectivamente.

Com isso, a sirene é devidamente acionada ou desativada quando necessário.

5.4 Acesso ao Alarme por meio do Telefone Celular

O acesso ao alarme por meio do telefone celular é garantido pelo software mobile web site interface. Este por sua vez foi desenvolvido com a tecnologia .Net em C# e ASP.NET. Estas tecnologias facilitaram bastante a confecção do site uma vez que utilizam a programação orientada a objetos e disponibilizam componentes que fornecem suporte a telefones celulares.

Com isso exposto, fica evidente que a utilização do telefone celular para o controle e monitoramento do alarme ocorre de forma segura e eficiente.

5.5 Considerações Finais

Esta subseção contém as considerações finais sobre o trabalho. As mesmas estão divididas em três subseções: testes realizados, dificuldades encontradas e conclusões.

5.5.1 Testes Realizados

Os testes realizados portaram-se satisfatoriamente. A figura 5-1 mostra a imagem de todo o sistema que é acessado e controlado pelo celular.

Figura 5-1 - Sistema de alarme controlado pelo celular

Quando o sistema está ativo e existe uma detecção de movimento, ele aguarda dois segundos, captura a imagem monitorada e envia uma mensagem ao telefone celular para avisar que o alarme está disparado. Ao receber a mensagem, a qual demora em média de vinte a trinta segundos, o usuário acessa o sistema de alarme por meio do telefone e tem acesso a foto obtida no momento do disparo do alarme. Deste modo, caso o usuário detecte algum estranho no ambiente monitorado, ele pode avisar a polícia imediatamente e informar o ocorrido. Caso contrário, ele pode desativar o alarme e ativá-lo novamente aguardando um novo evento.

Para que o usuário possa controlar o sistema de alarme é preciso que ele informe o nome e senha de login conforme a figura 5-2.

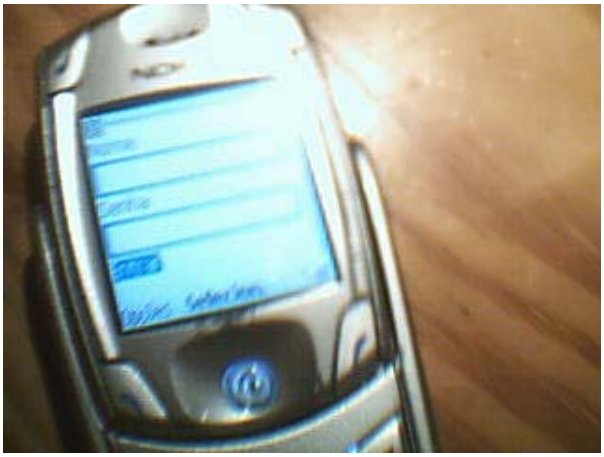


Figura 5-2 Acesso ao sistema através do telefone celular

Uma vez logado no sistema, é possível ativar ou desativar o alarme conforme figura 5-3.

Figura 5-3 Controle do alarme por meio do telefone celular

Ainda, sempre que o sistema estiver ativo e for acessado pelo telefone celular, é possível ver a foto atual do ambiente monitorado. Isto deve-se ao fato do software controle e monitoramento obter uma foto do ambiente a cada minuto e deixá-la disponível para monitoramento.

5.5.2 Dificuldades Encontradas

Uma das dificuldades encontradas diz respeito a comunicação entre os softwares *controle e monitoramento* e o *mobile web site interface.* Inicialmente foi utilizado um arquivo XML. Porém, como os softwares o acessavam ao mesmo tempo, não foi possível a sua utilização. Para a resolução desse problema foi utilizado um SGBD (Sistema de Gerenciamento de Bando de Dados) como descrito na subseção 4.5.

A outra dificuldade foi em relação à obtenção da foto no momento do disparo do alarme. Esse fato surgiu em decorrência de que o primeiro frame resultante do disparo possuía mudanças mínimas e não continha a imagem do intruso. Para isso, antes da obtenção do frame com a foto do mesmo, foi necessário aguardar dois segundos após o momento da detecção de movimento.

5.5.3 Conclusões

Neste trabalho, foi implementada e testada uma alternativa em segurança residencial muito eficaz e confiável.

O método utilizado para detecção de movimento através de análise de imagens mostrou-se um meio eficiente e econômico em substituição aos sensores mecânicos, eletromagnéticos, térmicos ou acústicos. Isso deve-se ao fato de que a mesma câmera utilizada para a obtenção da foto é utilizada também para a detecção de movimento e assim, não há necessidade de aquisição de um outro equipamento para este fim.

O envio da mensagem utilizando garantia de entrega implementado no software controle e monitoramento portou-se bem e atendeu ao escopo deste trabalho. Uma vez que a entrega da mensagem deve ser assegurada.

Ainda, a possibilidade de controlar e monitorar o alarme da residência por meio do telefone celular apresentou uma solução inteligente e necessária para os dias atuais.

Com tudo isso, esse projeto torna uma poderosa ferramenta para a segurança residencial apresentando maior segurança e tranqüilidade aos cidadãos.

REFERÊNCIAS

Connection Software **[Home Page]** 2007 Disponível em https://www.csoft.co.uk

E. STRINGA, S. Regazzoni, Real-Time Video-Shot Detection for Scene Surveillance Applications IEEE Transactions on Image Processing, Vol. 9, no. 1, 2000: p. 69-79.

MALVINO, Albert Paul. **Eletrônica: volume 1.** 4. ed. São Paulo: Makron Books, 1995.

MySql **[Home Page]** 2007 Disponível em http://www.mysqlbrasil.com.br

Rogercom **[Home Page].** 2007. Disponível em: <http://www.rogercom.com>.

SOARES, André Borin; Figueiró, Thiago; Susin, Altamiro. Artigo: *Caracterização do desempenho de métodos de detecção de movimento aplicado a localização de pessoas através de visão computacional*. Inst. de Informática - UFRGS , 2004: p. 1.
Disponível em: http://www.lapsi.eletro.ufrgs.br/~figueiro/SIDEEmov.pdf
Acesso em 20/10/2007

TAKAHASHI, Kazuhiko; SAKAGUCHI, Tatsumi; OHYA, Jun. Real-time estimation of human body postures using Kalman filter. In: INTERNATIONAL WORKSHOP ON ROBOT AND HUMAN INTERACTION, 8. 1999, Roma: 1999. p. 189-194. Disponível em:
http://www.mic.atr.co.jp/~tatsu/research/publication/pubtatsu.html
Acesso em: 25/10/2007

WHATIS. The Leading IT Encyclopedia and Learning Center.
Disponível em: http://whatis.techtarget.com/definition/0,,sid9_gci212900,00.html

Acesso em: 10/10/2007.

YOUNG, I.T.; Gerbrands, J.J.; Van Vilet, L.J. Fundamentals of Image Processing. - Quantitative Imaging Group - Department of Imaging Science and Technology, Faculty of Applied Sciences. Delft University of Technology – Delft The Netherlands
Disponível em: http://www.ph.tn.tudelft.nl/Courses/FIP/noframes/fip.html
Acesso em: 15/09/2007

APÊNDICE A – CÓDIGO FONTE DO SOFTWARE CONTROLE E MONITORAMENTO

Pacote Detecção de Movimento

ProtecaoResidencial.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using MySQLDriverCS;
using motion;
using Mysql;
using Sirene;
using CSoft.SmsService;

namespace ProtecaoResidencial
{

    public partial class Main : Form
    {
        public bool ativado = false;
        public Main()
        {
            InitializeComponent();
        }

     private void desligacamera()
     {
       Imagem.DesligarCamera();
     }

     private void ligaCamera()
        {

          Imagem.LigarCamera(Imagem.CapturarDispositivo());

        }


     private void Main_Load(object sender, EventArgs e)
        {
            // inicia o sistema com o alarme desativado
```

```
            ligaCamera();
            Conexao.UpdateStatus("2");
            Conexao.UpdateAcao("2");
            Procedimento.DeligaSirene();
        }

        private void timerVigia_Tick(object sender, EventArgs e)
        {

            if (Conexao.acionar())    //verifica se o sistema foi Ativado(1), desativado(2) ou
disparado(3)
                {
                   string acao = Conexao.SelectAcao();
                    if (acao.Trim() == "1")
                    {
                        ativarAlarme();
                        ativado = true;
                    }
                    else if (acao.Trim() == "2")
                    {
                        ativado = false;
                        desativarAlarme();
                    }
                    else
                    {
                        ativado = false;
                        disparar();
                    }
                }

        }

        private void ativarAlarme()
         {
             //coloca o status do sistema em ativado
             Conexao.UpdateStatus("1");

             //ativa a câmera
            Imagem.StatusProtecao(true);

         }

        private void desativarAlarme()
        {
            // coloca o status do sistema em desativado
            Conexao.UpdateStatus("2");


            //desativa a câmera
            Imagem.StatusProtecao(false);
```

```
            //desliga a sirene
            Procedimento.DeligaSirene();
        }

        private void disparar()
        {
            //coloca o status do sistema em desativado
            Conexao.UpdateStatus("3");


            System.Threading.Thread.Sleep(2000); // aguarda 2 segundos para a obtenção da
imagem
            Imagem.CapturarImagem();
            Conexao.UpdateDataFoto();
            Imagem.StatusProtecao(false);

            //liga a sirene
            Procedimento.LigaSirene();

            //envio da mensagem
            Mensagem.EnviarSMS("O seu alarme disparou em "+ DateTime.Now.ToShortDateString() +
" às " + DateTime.Now.ToShortTimeString() + ". Acesse sua página para monitoramento!!!");
        }

        private void timerRelogioPainel_Tick(object sender, EventArgs e)
        {
            Relogio.Text = DateTime.Now.ToLongTimeString();
        }

        private void TimerFoto_Tick(object sender, EventArgs e)
        {
            if (ativado)
            {
               Imagem.CapturarImagem();
               Conexao.UpdateDataFoto();
            }
        }

    }
}
```

Camera.cs

```
namespace motion
{
        using System;
        using System.Drawing;
        using System.Threading;
        using VideoSource;
    using System.Windows.Forms;

        /// <summary>
        /// Camera class
        /// </summary>
        public class Camera
```

```
{
        private IVideoSource  videoSource = null;
        private IMotionDetector       motionDetecotor = null;
        private Bitmap                lastFrame = null;
 private Bitmap          currentFrame = null;

        // image width and height
        private int           width = -1, height = -1;

        // alarm level
 private double alarmLevel = 0.020; //0.005;

        //
        public event EventHandler     NewFrame;
        public event EventHandler     Alarm;

        // LastFrame property
        public Bitmap LastFrame
        {
                get { return lastFrame; }
        }

 // LastFrame property
 public Bitmap CurrentFrame
 {
     get { return currentFrame; }
 }
        // Width property
        public int Width
        {
                get { return width; }
        }
        // Height property
        public int Height
        {
                get { return height; }
        }
        // FramesReceived property
        public int FramesReceived
        {
                get { return ( videoSource == null ) ? 0 : videoSource.FramesReceived; }
        }
        // BytesReceived property
        public int BytesReceived
        {
                get { return ( videoSource == null ) ? 0 : videoSource.BytesReceived; }
        }
        // Running property
        public bool Running
        {
                get { return ( videoSource == null ) ? false : videoSource.Running; }
        }
        // MotionDetector property
        public IMotionDetector MotionDetector
        {
                get { return motionDetecotor; }
                set { motionDetecotor = value; }
        }

        // Constructor
        public Camera( IVideoSource source ) : this( source, null )
        { }
        public Camera( IVideoSource source, IMotionDetector detector )
        {
                this.videoSource = source;
                this.motionDetecotor = detector;
                videoSource.NewFrame += new CameraEventHandler( video_NewFrame );
        }

        // Start video source
        public void Start( )
        {
                if ( videoSource != null )
                {
                        videoSource.Start( );
                }
        }
```

```
// Siganl video source to stop
public void SignalToStop( )
{
	if ( videoSource != null )
	{
		videoSource.SignalToStop( );
	}
}

// Wait video source for stop
public void WaitForStop( )
{
	// lock
	Monitor.Enter( this );

	if ( videoSource != null )
	{
		videoSource.WaitForStop( );
	}
	// unlock
	Monitor.Exit( this );
}

// Abort camera
public void Stop( )
{
	// lock
	Monitor.Enter( this );

	if ( videoSource != null )
	{
		videoSource.Stop( );
	}
	// unlock
	Monitor.Exit( this );
}

// Lock it
public void Lock( )
{
	Monitor.Enter( this );
}

// Unlock it
public void Unlock( )
{
	Monitor.Exit( this );
}

// On new frame
private void video_NewFrame( object sender, CameraEventArgs e )
{
	try
	{
		// lock
		Monitor.Enter( this );

		// dispose old frame
		if ( lastFrame != null )
		{
			lastFrame.Dispose( );
		}

		lastFrame = (Bitmap) e.Bitmap.Clone( );

		// apply motion detector
 if (motionDetecotor != null)
 {
     motionDetecotor.ProcessFrame(ref lastFrame);

     // check motion level
     if (motionDetecotor.MotionLevel >= alarmLevel)
     {
         MotionDetector = null;
         currentFrame = (Bitmap)e.Bitmap.Clone();
         // Alarm(this, new EventArgs());
```

```
                        Imagem.DispararAlarem();
                    }
                }

                        // image dimension
                        width = lastFrame.Width;
                        height = lastFrame.Height;
                    }
                    catch ( Exception )
                    {
                    }
                    finally
                    {
                        // unlock
                        Monitor.Exit( this );
                    }

                    // notify client
                    if ( NewFrame != null )
                        NewFrame( this, new EventArgs( ) );
                }
            }
}
```

Imagem.cs

```
namespace motion
{
    public class Imagem
    {

        private static IMotionDetector detector = new MotionDetector1( );
        private static int detectorType = 1;
        private static Camera camera;
        private static motion.CameraWindow cameraWindow;



        public static IVideoSource CapturarDispositivo()
        {
            CaptureDevice localSource = new CaptureDevice();
              try
            {
                FilterCollection filters = new
FilterCollection(FilterCategory.VideoInputDevice);
                if (filters.Count == 0)
                {
                    throw new ApplicationException();
                }

                // create video source

                localSource.VideoSource = filters[0].MonikerString;
                //OpenVideoSource(localSource);
                return localSource;

            }
            catch (Exception ex)
            {
                throw new Exception("Erro ao selecionar o dispositivo", ex);
            }
        }

        public static void DesligarCamera()
        {
            camera.Stop();
        }

        public static void LigarCamera(IVideoSource source)
         {

            // Ativa ou desativa a detecção da câmera
            if (detector != null)
```

```
              {
                detector.MotionLevelCalculation = false;
              }

            // create camera
            camera = new Camera(source, detector);
            // start camera
            camera.Start();
         }

        public static void StatusProtecao(bool status)
        {
            detector.MotionLevelCalculation = status;
        }

        public static void CapturarImagem()  // este método é chamada quando a câmera detecta
um movimento
        {
            //camera.CurrentFrame.Save("C:/Inetpub/wwwroot/Residencia/foto/Foto.png");
            camera.LastFrame.Save("C:/Inetpub/wwwroot/Residencia/foto/Foto.png");
        }

        public static void DispararAlarem()  // este método é chamada quando a câmera detecta
um movimento
        {
            //detector.MotionLevelCalculation = false;
            Conexao.UpdateAcao("3");

        }

        public static void SetMotionDetector()
        {

            // enable/disable motion alarm
            if (detector != null)
            {
                detector.MotionLevelCalculation = false;
            }

            // set motion detector to camera
            if (camera != null)
            {
                camera.Lock();
                camera.MotionDetector = detector;
                camera.Unlock();
            }
        }
    }
}
```

IMotionDetector.cs

```
namespace motion
{
       using System;
       using System.Drawing;

       /// <summary>
       /// IMotionDetector interface
       /// </summary>
       public interface IMotionDetector
       {
              /// <summary>
              /// Motion level calculation - calculate or not motion level
              /// </summary>
              bool MotionLevelCalculation{ set; get; }

              /// <summary>
              /// Motion level - amount of changes in percents
              /// </summary>
              double MotionLevel{ get; }

              /// <summary>
              /// Process new frame
              /// </summary>
              void ProcessFrame( ref Bitmap image );
```

```
		/// <summary>
		/// Reset detector to initial state
		/// </summary>
		void Reset( );
	}
}
```

MotionDetector1.cs

```
namespace motion
{
	using System;
	using System.Drawing;
	using System.Drawing.Imaging;

	using AForge.Imaging;
	using AForge.Imaging.Filters;

	/// <summary>
	/// MotionDetector1
	/// </summary>
	public class MotionDetector1 : IMotionDetector
	{
		private IFilter	grayscaleFilter = new GrayscaleBT709( );
		private Difference differenceFilter = new Difference( );
	 private Threshold thresholdFilter = new Threshold( 15 );
		private IFilter erosionFilter = new Erosion( );
		private Merge mergeFilter = new Merge( );

		private IFilter extrachChannel = new ExtractChannel( RGB.R );
		private ReplaceChannel replaceChannel = new ReplaceChannel( RGB.R, null );

		private Bitmap backgroundFrame;
	 private BitmapData bitmapData;

		private bool   calculateMotionLevel = false;
		private int		width;  // image width
		private int		height; // image height
		private int		pixelsChanged;

		// Motion level calculation - calculate or not motion level
		public bool MotionLevelCalculation
		{
			get { return calculateMotionLevel; }
			set { calculateMotionLevel = value; }
		}

		// Motion level - amount of changes in percents
		public double MotionLevel
		{
			get { return (double) pixelsChanged / ( width * height ); }
		}


		// Constructor
		public MotionDetector1( )
		{
		}

		// Reset detector to initial state
		public void Reset( )
		{
			if ( backgroundFrame != null )
			{
				backgroundFrame.Dispose( );
				backgroundFrame = null;
			}
```

```
    }

    // Process new frame
    public void ProcessFrame( ref Bitmap image )
    {
            if ( backgroundFrame == null )
            {
                    // create initial backgroung image
                    backgroundFrame = grayscaleFilter.Apply( image );

                    // get image dimension
                    width  = image.Width;
                    height = image.Height;

                    // just return for the first time
                    return;
            }

            Bitmap tmpImage;

            // apply the grayscale file
            tmpImage = grayscaleFilter.Apply( image );

            // set backgroud frame as an overlay for difference filter
            differenceFilter.OverlayImage = backgroundFrame;

// apply difference filter
Bitmap tmpImage2 = differenceFilter.Apply( tmpImage );

// lock the temporary image and apply some filters on the locked data
bitmapData = tmpImage2.LockBits( new Rectangle( 0, 0, width, height ),
    ImageLockMode.ReadWrite, PixelFormat.Format8bppIndexed );

// threshold filter
thresholdFilter.ApplyInPlace( bitmapData );
// erosion filter
Bitmap tmpImage3 = erosionFilter.Apply( bitmapData );

// unlock temporary image
tmpImage2.UnlockBits( bitmapData );
tmpImage2.Dispose( );

// calculate amount of changed pixels
            pixelsChanged = ( calculateMotionLevel ) ?
                    CalculateWhitePixels( tmpImage3 ) : 0;

            // dispose old background
            backgroundFrame.Dispose( );
            // set backgound to current
            backgroundFrame = tmpImage;

            // extract red channel from the original image
            Bitmap redChannel = extrachChannel.Apply( image );

            //  merge red channel with moving object
            mergeFilter.OverlayImage = tmpImage3;
            Bitmap tmpImage4 = mergeFilter.Apply( redChannel );
            redChannel.Dispose( );
            tmpImage3.Dispose( );

            // replace red channel in the original image
            replaceChannel.ChannelImage = tmpImage4;
            Bitmap tmpImage5 = replaceChannel.Apply( image );
            tmpImage4.Dispose( );

            image.Dispose( );
            image = tmpImage5;
    }

    // Calculate white pixels
    private int CalculateWhitePixels( Bitmap image )
    {
            int count = 0;

            // lock difference image
            BitmapData data = image.LockBits( new Rectangle( 0, 0, width, height ),
                    ImageLockMode.ReadOnly, PixelFormat.Format8bppIndexed );
```

```
                    int offset = data.Stride - width;

                    unsafe
                    {
                           byte * ptr = (byte *) data.Scan0.ToPointer( );

                           for ( int y = 0; y < height; y++ )
                           {
                                  for ( int x = 0; x < width; x++, ptr++ )
                                  {
                                         count += ( (*ptr) >> 7 );
                                  }
                                  ptr += offset;
                           }
                    }
                    // unlock image
                    image.UnlockBits( data );

                    return count;
             }
       }
}
```

Envio de Mensagem (SMS)

Mensagem.cs

```
namespace CSoft.SmsService
{
    public class Mensagem
    {
        public static void EnviarSMS(string msg)
        {
            try
             {
                 Recipient recipients = new Recipient();
                 CSoft.SmsService.Message messageContainer = new CSoft.SmsService.Message();
                 string smsMessageID = string.Empty;
                 Reserved reserved = new Reserved();
                 string infoText;
                 string libraryNo;
                 string result;

                 recipients.SendTo = "556181579910";
                 messageContainer.TextMessage = msg;
                 Service service = new Service();
                 result = service.SubmitMessage("GFaria.211590", "81931748 ", recipients,
messageContainer, null,
                 ref smsMessageID, ref reserved, out infoText, out libraryNo);

                 //MessageBox.Show(string.Format("Result code of {0} was returned." +
Environment.NewLine + Environment.NewLine +
                 //    "Message was '{1}'." + Environment.NewLine + Environment.NewLine +
"CSoft message ID is '{2}'.",
                 //    result, infoText, smsMessageID), "Information", MessageBoxButtons.OK,
MessageBoxIcon.Information);

             }
             catch (Exception ex)
              {
                throw new Exception("Erro ao mandar a mensagem SMS", ex);
              }
        }
    }
}
```

CSoftSmsService.cs

```
namespace CSoft.SmsService
{
    using System.Diagnostics;
    using System.Xml.Serialization;
    using System;
    using System.Web.Services.Protocols;
    using System.ComponentModel;
    using System.Web.Services;



    [System.Diagnostics.DebuggerStepThroughAttribute()]
    [System.ComponentModel.DesignerCategoryAttribute("code")]
    [System.Web.Services.WebServiceBindingAttribute(Name="Service",
Namespace="http://www.csoft.co.uk/dtd/sendsms5.wsdl")]
    public class Service : System.Web.Services.Protocols.SoapHttpClientProtocol {

        /// <remarks/>
        public Service() {
            this.Url = "http://www.csoft.co.uk/sendsms5";
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapDocumentMethodAttribute("",
RequestNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
ResponseNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        [return: System.Xml.Serialization.XmlElementAttribute("Report",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
        public string
SubmitMessage([System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaFo
rm.Unqualified, IsNullable=true)] string Username,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string PIN,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] Recipient Recipient,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] Message Message,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string ReplyTo,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] ref string MessageIdentifier,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] ref Reserved Reserved,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] out string Text,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="positiveInteger")] out string LibraryNumber) {
            object[] results = this.Invoke("SubmitMessage", new object[] {
                        Username,
                        PIN,
                        Recipient,
                        Message,
                        ReplyTo,
                        MessageIdentifier,
                        Reserved});
            MessageIdentifier = ((string)(results[1]));
            Reserved = ((Reserved)(results[2]));
            Text = ((string)(results[3]));
            LibraryNumber = ((string)(results[4]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginSubmitMessage(string Username, string PIN, Recipient
Recipient, Message Message, string ReplyTo, string MessageIdentifier, Reserved Reserved,
System.AsyncCallback callback, object asyncState) {
```

```
            return this.BeginInvoke("SubmitMessage", new object[] {
                        Username,
                        PIN,
                        Recipient,
                        Message,
                        ReplyTo,
                        MessageIdentifier,
                        Reserved}, callback, asyncState);
        }

        /// <remarks/>
        public string EndSubmitMessage(System.IAsyncResult asyncResult, out string
MessageIdentifier, out Reserved Reserved, out string Text, out string LibraryNumber) {
            object[] results = this.EndInvoke(asyncResult);
            MessageIdentifier = ((string)(results[1]));
            Reserved = ((Reserved)(results[2]));
            Text = ((string)(results[3]));
            LibraryNumber = ((string)(results[4]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapDocumentMethodAttribute("",
RequestNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
ResponseNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        [return: System.Xml.Serialization.XmlElementAttribute("Text",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified, IsNullable=true)]
        public string
SubmitDeliveryReceipt([System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.Xml
SchemaForm.Unqualified, IsNullable=true)] string Username,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string PIN,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] System.DateTime Timestamp1,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string SentTo,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] ref string MessageIdentifier,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] ref int Report) {
            object[] results = this.Invoke("SubmitDeliveryReceipt", new object[] {
                        Username,
                        PIN,
                        Timestamp1,
                        SentTo,
                        MessageIdentifier,
                        Report});
            MessageIdentifier = ((string)(results[1]));
            Report = ((int)(results[2]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginSubmitDeliveryReceipt(string Username, string PIN,
System.DateTime Timestamp1, string SentTo, string MessageIdentifier, int Report,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("SubmitDeliveryReceipt", new object[] {
                        Username,
                        PIN,
                        Timestamp1,
                        SentTo,
                        MessageIdentifier,
                        Report}, callback, asyncState);
        }

        /// <remarks/>
        public string EndSubmitDeliveryReceipt(System.IAsyncResult asyncResult, out string
MessageIdentifier, out int Report) {
            object[] results = this.EndInvoke(asyncResult);
            MessageIdentifier = ((string)(results[1]));
            Report = ((int)(results[2]));
            return ((string)(results[0]));
        }

        /// <remarks/>
```

```
        [System.Web.Services.Protocols.SoapDocumentMethodAttribute("",
RequestNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
ResponseNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        [return: System.Xml.Serialization.XmlElementAttribute("MessageIdentifier",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
        public string
DeliverMessage([System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaF
orm.Unqualified, IsNullable=true)] string Username,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string PIN,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] Recipient Recipient,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] Message Message,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string ReplyTo,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] System.DateTime Timestamp1,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] [System.Xml.Serialization.XmlIgnoreAttribute()] bool Timestamp1Specified,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] out int Report,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] out string Text) {
            object[] results = this.Invoke("DeliverMessage", new object[] {
                        Username,
                        PIN,
                        Recipient,
                        Message,
                        ReplyTo,
                        Timestamp1,
                        Timestamp1Specified});
            Report = ((int)(results[1]));
            Text = ((string)(results[2]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginDeliverMessage(string Username, string PIN, Recipient
Recipient, Message Message, string ReplyTo, System.DateTime Timestamp1, bool
Timestamp1Specified, System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("DeliverMessage", new object[] {
                        Username,
                        PIN,
                        Recipient,
                        Message,
                        ReplyTo,
                        Timestamp1,
                        Timestamp1Specified}, callback, asyncState);
        }

        /// <remarks/>
        public string EndDeliverMessage(System.IAsyncResult asyncResult, out int Report, out
string Text) {
            object[] results = this.EndInvoke(asyncResult);
            Report = ((int)(results[1]));
            Text = ((string)(results[2]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapDocumentMethodAttribute("",
RequestNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
ResponseNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        [return: System.Xml.Serialization.XmlElementAttribute("Messages",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
        public string
AvailableMessages([System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSche
maForm.Unqualified, IsNullable=true)] string Username,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string PIN,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] out Reserved Reserved) {
```

```
            object[] results = this.Invoke("AvailableMessages", new object[] {
                        Username,
                        PIN});
            Reserved = ((Reserved)(results[1]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginAvailableMessages(string Username, string PIN,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("AvailableMessages", new object[] {
                        Username,
                        PIN}, callback, asyncState);
        }

        /// <remarks/>
        public string EndAvailableMessages(System.IAsyncResult asyncResult, out Reserved
Reserved) {
            object[] results = this.EndInvoke(asyncResult);
            Reserved = ((Reserved)(results[1]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapDocumentMethodAttribute("",
RequestNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
ResponseNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        [return: System.Xml.Serialization.XmlElementAttribute("Currency",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
        public string
AvailableCredit([System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchema
Form.Unqualified, IsNullable=true)] string Username,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string PIN,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] out string Credit,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] out Reserved Reserved) {
            object[] results = this.Invoke("AvailableCredit", new object[] {
                        Username,
                        PIN});
            Credit = ((string)(results[1]));
            Reserved = ((Reserved)(results[2]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginAvailableCredit(string Username, string PIN,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("AvailableCredit", new object[] {
                        Username,
                        PIN}, callback, asyncState);
        }

        /// <remarks/>
        public string EndAvailableCredit(System.IAsyncResult asyncResult, out string Credit,
out Reserved Reserved) {
            object[] results = this.EndInvoke(asyncResult);
            Credit = ((string)(results[1]));
            Reserved = ((Reserved)(results[2]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapDocumentMethodAttribute("",
RequestNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
ResponseNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        [return: System.Xml.Serialization.XmlElementAttribute("Version",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
        public string
MobileMessengerVersion([System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.Xm
lSchemaForm.Unqualified)] out string ReleaseNotice,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
```

```
)] out string DownloadURL,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] out System.DateTime ReleaseDate) {
            object[] results = this.Invoke("MobileMessengerVersion", new object[0]);
            ReleaseNotice = ((string)(results[1]));
            DownloadURL = ((string)(results[2]));
            ReleaseDate = ((System.DateTime)(results[3]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginMobileMessengerVersion(System.AsyncCallback callback,
object asyncState) {
            return this.BeginInvoke("MobileMessengerVersion", new object[0], callback,
asyncState);
        }

        /// <remarks/>
        public string EndMobileMessengerVersion(System.IAsyncResult asyncResult, out string
ReleaseNotice, out string DownloadURL, out System.DateTime ReleaseDate) {
            object[] results = this.EndInvoke(asyncResult);
            ReleaseNotice = ((string)(results[1]));
            DownloadURL = ((string)(results[2]));
            ReleaseDate = ((System.DateTime)(results[3]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapDocumentMethodAttribute("",
RequestNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
ResponseNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        [return: System.Xml.Serialization.XmlElementAttribute("Report",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
        public int
CollectMessages([System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchema
Form.Unqualified, IsNullable=true)] string Username,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string PIN,
[System.Xml.Serialization.XmlArrayAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
[System.Xml.Serialization.XmlArrayItemAttribute("item",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)] out MOMessage[] MessageList) {
            object[] results = this.Invoke("CollectMessages", new object[] {
                        Username,
                        PIN});
            MessageList = ((MOMessage[])(results[1]));
            return ((int)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginCollectMessages(string Username, string PIN,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("CollectMessages", new object[] {
                        Username,
                        PIN}, callback, asyncState);
        }

        /// <remarks/>
        public int EndCollectMessages(System.IAsyncResult asyncResult, out MOMessage[]
MessageList) {
            object[] results = this.EndInvoke(asyncResult);
            MessageList = ((MOMessage[])(results[1]));
            return ((int)(results[0]));
        }

        /// <remarks/>
        [System.Web.Services.Protocols.SoapDocumentMethodAttribute("",
RequestNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
ResponseNamespace="http://www.csoft.co.uk/dtd/sendsms5.xsd",
Use=System.Web.Services.Description.SoapBindingUse.Literal,
ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
        [return: System.Xml.Serialization.XmlElementAttribute("PermittedReplyTo",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
        public string
PermittedReplyTo([System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchem
aForm.Unqualified, IsNullable=true)] string Username,
```

```
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)] string PIN,
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)] out Reserved Reserved) {
            object[] results = this.Invoke("PermittedReplyTo", new object[] {
                        Username,
                        PIN});
            Reserved = ((Reserved)(results[1]));
            return ((string)(results[0]));
        }

        /// <remarks/>
        public System.IAsyncResult BeginPermittedReplyTo(string Username, string PIN,
System.AsyncCallback callback, object asyncState) {
            return this.BeginInvoke("PermittedReplyTo", new object[] {
                        Username,
                        PIN}, callback, asyncState);
        }

        /// <remarks/>
        public string EndPermittedReplyTo(System.IAsyncResult asyncResult, out Reserved
Reserved) {
            object[] results = this.EndInvoke(asyncResult);
            Reserved = ((Reserved)(results[1]));
            return ((string)(results[0]));
        }
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class Recipient {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string SendTo;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string SendCc;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string SendBcc;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string SendToAddressBook;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string SendCcAddressBook;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string SendBccAddressBook;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string PhoneMake;

        /// <remarks/>
```

```
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string PhoneModel;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public int MCC;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool MCCSpecified;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public int MNC;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool MNCSpecified;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public SendOptions SendOptions;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class SendOptions {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public System.DateTime SendTimeAbsolute;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool SendTimeAbsoluteSpecified;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public System.UInt16 SendTimeDelay;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool SendTimeDelaySpecified;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public DeliveryOption DeliveryOption;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool DeliveryOptionSpecified;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public bool DisableRepurposing;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool DisableRepurposingSpecified;
```

```
        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public bool DisableWalledGardenRule;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool DisableWalledGardenRuleSpecified;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public enum DeliveryOption {

        /// <remarks/>
        Default,

        /// <remarks/>
        Premier,

        /// <remarks/>
        LeastCost,

        /// <remarks/>
        Economy,
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class MOMessage {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public string SendTo;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public string ReplyTo;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public System.DateTime Timestamp1;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public Message Message;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class Message {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string TextMessage;

        /// <remarks/>
```

```
[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string FlashMessage;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public SmartMessage Ringtone;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public SmartMessage OperatorLogo;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public SmartMessage PictureMessage;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public SmartMessage GroupGraphic;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public SMSSUBMITPDU SmsSubmitPdu;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public WAPPush WAPPush;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public MMSNotification MMSNotification;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class SmartMessage {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="positiveInteger")]
        public string LibraryNumber;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="hexBinary", IsNullable=true)]
        public System.Byte[] Data;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public DataFormat Format;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool FormatSpecified;
    }
```

```
    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public enum DataFormat {

        /// <remarks/>
        [System.Xml.Serialization.XmlEnumAttribute("FORMAT-UNDEFINED")]
        FORMATUNDEFINED,

        /// <remarks/>
        RTTTL,

        /// <remarks/>
        OTA,
    }

    /// <remarks/>
    [System.Xml.Serialization.XmlTypeAttribute(TypeName="SMS-SUBMIT-PDU",
Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd")]
    public class SMSSUBMITPDU {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="hexBinary")]
        public System.Byte[] DataCodingScheme;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="hexBinary")]
        public System.Byte[] ProtocolIdentifier;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="hexBinary")]
        public System.Byte[] UserDataHeader;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="hexBinary")]
        public System.Byte[] UserData;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class WAPPush {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public SimplePush SimplePush;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public MMSLibraryNumber LibraryNumber;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public MMSFileUpload FileUpload;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public si si;
    }
```

```
    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class SimplePush {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="positiveInteger")]
        public string LibraryNumber;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string TextMessage;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class MMSLibraryNumber {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="positiveInteger")]
        public string LibraryNumber;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string TextMessage;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class MMSFileUpload {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string FileName1;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="hexBinary", IsNullable=true)]
        public System.Byte[] FileData1;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string FileType1;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string FileTitle1;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string ContentType1;
    }

    /// <remarks/>
```

```
[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class si {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public indication indication;

        /// <remarks/>

[System.Xml.Serialization.XmlArrayAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
        [System.Xml.Serialization.XmlArrayItemAttribute(typeof(item),
Form=System.Xml.Schema.XmlSchemaForm.Unqualified, IsNullable=false)]
        public item[] info;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class indication {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="anyURI")]
        public string href;

        /// <remarks/>
        [System.Xml.Serialization.XmlElementAttribute("si-id",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified, IsNullable=true)]
        public string siid;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public System.DateTime created;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool createdSpecified;

        /// <remarks/>
        [System.Xml.Serialization.XmlElementAttribute("si-expires",
Form=System.Xml.Schema.XmlSchemaForm.Unqualified)]
        public System.DateTime siexpires;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool siexpiresSpecified;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public action action;

        /// <remarks/>
        [System.Xml.Serialization.XmlIgnoreAttribute()]
        public bool actionSpecified;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string content;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public enum action {
```

```
        /// <remarks/>
        [System.Xml.Serialization.XmlEnumAttribute("signal-medium")]
        signalmedium,

        /// <remarks/>
        [System.Xml.Serialization.XmlEnumAttribute("signal-none")]
        signalnone,

        /// <remarks/>
        [System.Xml.Serialization.XmlEnumAttribute("signal-low")]
        signallow,

        /// <remarks/>
        [System.Xml.Serialization.XmlEnumAttribute("signal-high")]
        signalhigh,

        /// <remarks/>
        [System.Xml.Serialization.XmlEnumAttribute("signal-delete")]
        signaldelete,
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class item {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
)]
        public string pcdata;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, DataType="NMTOKEN")]
        public string @class;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class MMSNotification {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public MMSLibraryNumber LibraryNumber;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public MMSFileUpload FileUpload;
    }

    /// <remarks/>

[System.Xml.Serialization.XmlTypeAttribute(Namespace="http://www.csoft.co.uk/dtd/sendsms5.xsd"
)]
    public class Reserved {

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string Field1;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string Field2;
```

```
        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string Field3;

        /// <remarks/>

[System.Xml.Serialization.XmlElementAttribute(Form=System.Xml.Schema.XmlSchemaForm.Unqualified
, IsNullable=true)]
        public string Field4;
    }
}
```

Conexão com o MySql

Conexao.cs

```
namespace Mysql
{
    public class Conexao
    {

        public static bool acionar()
        {
            try
            {
               string status = "";
               string acao = "";
               MySQLConnection conselectMonitoramento;
               conselectMonitoramento = new MySQLConnection(new
MySQLConnectionString("localhost",
                                                                    "bd_alarme",
                                                                    "root",
                                                                    "").AsString);

                conselectMonitoramento.Open();

                //módulo select
                string sql = "select acao, status from alarme where Id_alarme =1";

                MySQLCommand cmd = new MySQLCommand(sql, conselectMonitoramento);
                MySQLDataReader reader = cmd.ExecuteReaderEx();
                while (reader.Read())
                {
                    status = reader.GetString(1).ToString().Trim();
                    acao = reader.GetString(0).ToString().Trim();
                }

                reader.Close();
                cmd.Dispose();
                //fim do módulo select
                conselectMonitoramento.Close();
                conselectMonitoramento.Dispose();

                if (status.Trim() == acao.Trim())
                 {
                   return false;
                 }
                   else
                    {
                      return true;
                    }
            }
            catch (Exception ex)
            {
                Exception tmpEx = new Exception("Erro ao buscar dados no banco", ex);
```

```
                throw tmpEx;
            }
        }

     public static string SelectAcao()
        {
           try
            {
                string acao = "";
                MySQLConnection conselectMonitoramento;
                conselectMonitoramento = new MySQLConnection(new
MySQLConnectionString("localhost",
                                                                "bd_alarme",
                                                                "root",
                                                                "").AsString);


               conselectMonitoramento.Open();

                //módulo select
                string sql = "select acao from alarme where Id_alarme =1";

                MySQLCommand cmd = new MySQLCommand(sql, conselectMonitoramento);
                MySQLDataReader reader = cmd.ExecuteReaderEx();
                while (reader.Read())
                       {
                          acao = reader.GetString(0).ToString().Trim();
                       }


                reader.Close();
                cmd.Dispose();
                //fim do módulo select
                conselectMonitoramento.Close();
                conselectMonitoramento.Dispose();

                return acao;

            }
            catch (Exception ex)
            {
                Exception tmpEx = new Exception("Erro ao buscar dados no banco", ex);
                throw tmpEx;
            }
        }

        public static void UpdateDataFoto()
        {
            string DT_Foto = DateTime.Now.Year + "-" + DateTime.Now.Month + "-" +
DateTime.Now.Day + " " + DateTime.Now.Hour + ":" + DateTime.Now.Minute + ":" +
DateTime.Now.Second;
            try
            {
                MySQLConnection conUpdateStatus;
                conUpdateStatus = new MySQLConnection(new MySQLConnectionString("localhost",
                                                                "bd_alarme",
                                                                "root",
                                                               "").AsString);
                conUpdateStatus.Open();

                //módulo insert
                MySQLCommand cmd = new MySQLCommand("UPDATE alarme SET Dt_foto= '" + DT_Foto +
"' WHERE Id_alarme=1", conUpdateStatus);
                int cnt = cmd.ExecuteNonQuery();

                //fim do módulo select
                conUpdateStatus.Close();
            }
            catch (Exception ex)
            {
                Exception tmpEx = new Exception("Erro ao atualizar a hora da foto no banco de
dados", ex);
                throw tmpEx;
            }
        }
        public static void UpdateStatus(string Vstatus)
        {
```

```
            try
            {
                MySQLConnection conUpdateStatus;
                conUpdateStatus = new MySQLConnection(new MySQLConnectionString("localhost",
                                                                                "bd_alarme",
                                                                                "root",
                                                                               "").AsString);
                conUpdateStatus.Open();

                //módulo insert
                MySQLCommand cmd = new MySQLCommand("UPDATE alarme SET status=" + Vstatus + "
WHERE Id_alarme=1", conUpdateStatus);
                int cnt = cmd.ExecuteNonQuery();

                //fim do módulo select
                conUpdateStatus.Close();
            }
            catch (Exception ex)
            {
                Exception tmpEx = new Exception("Erro ao atualizar o status do alarme no
banco", ex);
                throw tmpEx;
            }
        }
        public static void UpdateAcao(string Vacao)
        {

            try
            {
                MySQLConnection conUpdateAcao;
                conUpdateAcao = new MySQLConnection(new MySQLConnectionString("localhost",
                                                                                "bd_alarme",
                                                                                "root",
                                                                               "").AsString);
                conUpdateAcao.Open();

                //módulo insert
                MySQLCommand cmd = new MySQLCommand("UPDATE alarme SET acao=" + Vacao + "
WHERE Id_alarme=1", conUpdateAcao);
                int cnt = cmd.ExecuteNonQuery();

                //fim do módulo select
                conUpdateAcao.Close();
            }
            catch (Exception ex)
            {
                Exception tmpEx = new Exception("Erro ao atualizar a acao do alarme no banco",
ex);
                throw tmpEx;
            }
        }

    }


}
```

Acionamento da Sirene

Paralela.cs

```
namespace Sirene
{
    class Paralela
    {
        // Escreve um byte no endereço
        [DllImport("Inpout32.dll", EntryPoint = "Out32")]
        public static extern void Escrever(int endereco, byte valor);
```

```
        // Lê um byte do endereço
        [DllImport("Inpout32.dll", EntryPoint = "Inp32")]
        public static extern byte Ler(int endereco);
    }
}
```

Procedimento.cs

```
namespace Sirene
{
    public class Procedimento
    {

        public static void LigaSirene()
        {
            try
            {
                int endereco = Convert.ToInt32("378", 16);
                byte dados = Convert.ToByte("00000001", 2);
                Paralela.Escrever(endereco, dados);
            }
            catch (Exception ex)
            {
                throw new Exception("Erro ao ligar a sirene", ex);
            }
        }

        public static void DeligaSirene()
        {
            try
            {
                int endereco = Convert.ToInt32("378", 16);
                byte dados = Convert.ToByte("00000000", 2);
                Paralela.Escrever(endereco, dados);
            }
            catch (Exception ex)
            {
                throw new Exception("Erro ao desligar a sirene", ex);
            }
        }

        public static bool VerificaCabo()
        {
            try
            {
                bool CaboCortado = true;
                //byte A = Paralela.Ler(Convert.ToInt32("379", 16));
                //byte A = 10;
                //string c = Convert.ToString(A);
                //A = Convert.ToByte(c, 2);

                //txtLeitura.Text = Convert.ToString(A, 2);
                return CaboCortado;
            }
            catch (Exception ex)
            {
                throw new Exception("Erro ao verificar os cabos", ex);
            }
        }



    }
}
```

Program.cs

```
namespace Sirene
```

```
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

APÊNDICE B – CÓDIGO FONTE DO SOFTWARE MOBILE WEB SITE INTERFACE

Interface do Usuário

Login.aspx.cs

```
namespace Residencia
{
        /// <summary>
        /// Summary description for MobileWebForm1.
        /// </summary>
        public class login : System.Web.UI.MobileControls.MobilePage
        {
                protected System.Web.UI.MobileControls.Label lbltitulo;
                protected System.Web.UI.MobileControls.Label lblNome;
                protected System.Web.UI.MobileControls.TextBox txtNome;
                protected System.Web.UI.MobileControls.Label lblsenha;
                protected System.Web.UI.MobileControls.TextBox txtSenha;
                protected System.Web.UI.MobileControls.Command Command1;
                protected System.Web.UI.MobileControls.Form Form1;

                private void Page_Load(object sender, System.EventArgs e)
                {
                        // Put user code to initialize the page here
                }

                #region Web Form Designer generated code
                override protected void OnInit(EventArgs e)
                {
                        //
                        // CODEGEN: This call is required by the ASP.NET Web Form Designer.
                        //
                        InitializeComponent();
                        base.OnInit(e);
                }

                /// <summary>
                /// Required method for Designer support - do not modify
                /// the contents of this method with the code editor.
                /// </summary>
                private void InitializeComponent()
                {
                        this.Command1.Click += new System.EventHandler(this.Command1_Click);
                        this.Load += new System.EventHandler(this.Page_Load);
```

```
            }
            #endregion

            private void Command1_Click(object sender, System.EventArgs e)
            {
                  if (MySQL.Usuario(txtNome.Text.Trim(), txtSenha.Text.Trim()))  // faz o
login do usuário
                    {
                      RedirectToMobilePage("Monitoramento_Controle.aspx");
                    }


            }
      }
}
```

Monitoramento_Controle.aspx.cs

```
namespace Residencia
{
      /// <summary>
      /// Summary description for Monitoramento_Controle.
      /// </summary>
      public class Monitoramento_Controle : System.Web.UI.MobileControls.MobilePage
      {
            protected System.Web.UI.MobileControls.Label lblStatus;
            protected System.Web.UI.MobileControls.Command btnDesativar;
            protected System.Web.UI.MobileControls.Command btnAtivar;
            protected System.Web.UI.MobileControls.Form FrmPrincipal;
            protected System.Web.UI.MobileControls.Form FrmAguarde;
            protected System.Web.UI.MobileControls.Label lblAguarde;
            protected System.Web.UI.MobileControls.Label lblFoto;
            protected System.Web.UI.MobileControls.Image imagem;

            private void Page_Load(object sender, System.EventArgs e)
            {
                  carregar();
                  // Put user code to initialize the page here
       }

            #region Web Form Designer generated code
            override protected void OnInit(EventArgs e)
            {
                              //
                              // CODEGEN: This call is required by the ASP.NET Web Form
Designer.
                              //
            InitializeComponent();
            base.OnInit(e);
}

            /// <summary>
            /// Required method for Designer support - do not modify
            /// the contents of this method with the code editor.
            /// </summary>
            private void InitializeComponent()
            {
                  this.btnDesativar.Click += new
System.EventHandler(this.btnDesativar_Click);
                  this.btnAtivar.Click += new System.EventHandler(this.btnAtivar_Click);
                  this.Load += new System.EventHandler(this.Page_Load);

            }
            #endregion

            private void carregar()
            {
                  string horaFoto = MySQL.SelectData();
                  string status = MySQL.SelectStatus();
          btnAtivar.Visible = true;
                  btnDesativar.Visible = true;
```

```
            lblFoto.Text = "Foto obtida em: " +horaFoto;

                    switch ( status.Trim() )
                    {
                           case "1": // Sistema Ativado
                                  lblStatus.Text = "Sistema Ativado";
                                  btnAtivar.Visible = false;
                                  break;
                           case "2": // Sistema Desativado
                                  lblStatus.Text = "Sistema Desativado";
                                  btnDesativar.Visible = false;
                                  break;

                           case "3": // Sistema Disparado
                                  lblStatus.Text = "Sistema Disparado";
                                  btnAtivar.Visible = false;
                                  //imagem.Visible = true;
                                  break;
                    }

                    lblAguarde.Visible = false;

              }
              private void btnDesativar_Click(object sender, System.EventArgs e)
              {
              MySQL.UpdateAcao("2");

              while (MySQL.MudancaStatus()==true)
             {
                    btnDesativar.Visible = false;
                    lblAguarde.Visible = true;
                    System.Threading.Thread.Sleep(1000);
                }
                    carregar();


              }

              private void btnAtivar_Click(object sender, System.EventArgs e)
               {
                MySQL.UpdateAcao("1");

                    while (MySQL.MudancaStatus()== true)
                    {
                           btnAtivar.Visible = false;
                       lblAguarde.Visible = true;
                           System.Threading.Thread.Sleep(8000);
                    }
                    carregar();

               }
       }
}
```

Conexão com o MySql

MySQL.cs

```
namespace Residencia
{
       /// <summary>
       /// Summary description for MySQL.
       /// </summary>
       public class MySQL
       {
              public MySQL()
              {
                     //
                     // TODO: Add constructor logic here
                     //
```

```
		}

		public static bool MudancaStatus()
		{
			try
			{
				string status = "";
				string acao = "";
				MySQLConnection conselectMonitoramento;
				conselectMonitoramento = new MySQLConnection(new
MySQLConnectionString("localhost",
					"bd_alarme",
					"root",
					"").AsString);

				conselectMonitoramento.Open();

				//módulo select
				string sql = "select acao, status from alarme where Id_alarme
=1";

				MySQLCommand cmd = new MySQLCommand(sql,
conselectMonitoramento);
				MySQLDataReader reader = cmd.ExecuteReaderEx();
				while (reader.Read())
				{
					status = reader.GetString(1).ToString().Trim();
					acao = reader.GetString(0).ToString().Trim();
				}

				reader.Close();
				cmd.Dispose();
				//fim do módulo select
				conselectMonitoramento.Close();
				conselectMonitoramento.Dispose();

				if (status.Trim() == acao.Trim())
				{
					return false;
				}
				else
				{
					return true;
				}
			}
			catch (Exception ex)
			{
				Exception tmpEx = new Exception("Erro ao buscar dados no banco",
ex);
				throw tmpEx;
			}
		}

		public static void UpdateAcao(string Vacao)
		{

			try
			{
				MySQLConnection conUpdateAcao;
				conUpdateAcao = new MySQLConnection(new
MySQLConnectionString("localhost",
					"bd_alarme",
					"root",
					"").AsString);
				conUpdateAcao.Open();

				//módulo insert
				MySQLCommand cmd = new MySQLCommand("UPDATE alarme SET acao=" +
Vacao + " WHERE Id_alarme=1", conUpdateAcao);
				int cnt = cmd.ExecuteNonQuery();

				//fim do módulo select
				conUpdateAcao.Close();
			}
			catch (Exception ex)
			{
```

```
                        Exception tmpEx = new Exception("Erro ao atualizar a acao do
alarme no banco", ex);
                        throw tmpEx;
                }
        }


        public static string SelectStatus()
        {
                try
                {
                        string Status = "";
                        MySQLConnection conselectMonitoramento;
                        conselectMonitoramento = new MySQLConnection(new
MySQLConnectionString("localhost",
                                "bd_alarme",
                                "root",
                                "").AsString);


                        conselectMonitoramento.Open();

                        //módulo select
                        string sql = "select Status from alarme where Id_alarme =1";

                        MySQLCommand cmd = new MySQLCommand(sql,
conselectMonitoramento);
                        MySQLDataReader reader = cmd.ExecuteReaderEx();
                        while (reader.Read())
                        {
                                Status = reader.GetString(0).ToString().Trim();
                        }


                        reader.Close();
                        cmd.Dispose();
                        //fim do módulo select
                        conselectMonitoramento.Close();
                        conselectMonitoramento.Dispose();

                        return Status;

                }
                catch (Exception ex)
                {
                        Exception tmpEx = new Exception("Erro ao buscar dados no banco",
ex);
                        throw tmpEx;
                }
        }

        public static string SelectData()
        {
                try
                {
                        string Data = "";
                        MySQLConnection conselectMonitoramento;
                        conselectMonitoramento = new MySQLConnection(new
MySQLConnectionString("localhost",
                                "bd_alarme",
                                "root",
                                "").AsString);


                        conselectMonitoramento.Open();

                        //módulo select
                        string sql = "SELECT DATE_FORMAT(dt_foto, '%d/%m/%Y  -  %r' ) as
dt_foto FROM alarme where id_alarme = 1;";

                        MySQLCommand cmd = new MySQLCommand(sql,
conselectMonitoramento);
                        MySQLDataReader reader = cmd.ExecuteReaderEx();
                        while (reader.Read())
                        {
                                Data = reader.GetString(0).ToString().Trim();
                        }
```

```
                        reader.Close();
                        cmd.Dispose();
                        //fim do módulo select
                        conselectMonitoramento.Close();
                        conselectMonitoramento.Dispose();

                        return Data;

                }
                catch (Exception ex)
                {
                        Exception tmpEx = new Exception("Erro ao buscar dados no banco",
ex);
                        throw tmpEx;
                }
        }

        public static bool Usuario(string nome, string senha)
        {
                try
                {

                        bool UsuarioSistema ;
                        MySQLConnection conselectMonitoramento;
                        conselectMonitoramento = new MySQLConnection(new
MySQLConnectionString("localhost",
                                "bd_alarme",
                                "root",
                                "").AsString);


                        conselectMonitoramento.Open();

                        //módulo select
                        //string sql = "select nome, senha from usuario where nome =" +
nome+ " and senha ="+senha;
                string sql = "select nome from usuario where nome = '"+nome.Trim()+"' and
senha = '" + senha.Trim()+"'";
                        MySQLCommand cmd = new MySQLCommand(sql,
conselectMonitoramento);
                        MySQLDataReader reader = cmd.ExecuteReaderEx();
                        if (reader.Read())
                        {
                                UsuarioSistema = true;
                        }
                        else
                        {
                            UsuarioSistema = false;
                        }


                        reader.Close();
                        cmd.Dispose();
                        //fim do módulo select
                        conselectMonitoramento.Close();
                        conselectMonitoramento.Dispose();

                        return UsuarioSistema;

                }
                catch (Exception ex)
                {
                        Exception tmpEx = new Exception("Erro ao buscar dados no banco",
ex);
                        throw tmpEx;
                }
        }
    }
}
```