Fundação Educacional do Município de Assis
IMESA - Instituto Municipal de Ensino Superior de Assis

ROBERTA FERREIRA DE SOUZA

APLICATIVO ANDROID PARA MONITORAMENTO REMOTO POR CÂMERAS IP COM SENSOR DE PRESENÇA

Assis
2012

ROBERTA FERREIRA DE SOUZA

APLICATIVO ANDROID PARA MONITORAMENTO REMOTO POR CÂMERAS IP COM SENSOR DE PRESENÇA

Trabalho de Conclusão de Curso apresentado ao curso Bacharelado em Ciência da Computação do Instituto Municipal de Ensino Superior de Assis – IMESA e à Fundação Educacional do Município de Assis – FEMA, como requisito à obtenção do Certificado de Conclusão.

Orientador: Profº. Esp. Guilherme de Cleva Farto

Assis
2012

FICHA CATALOGRÁFICA

SOUZA, Roberta Ferreira de

Aplicativo Android para monitoramento remoto por câmeras IP com sensor de presença. Roberta Ferreira de Souza. Fundação Educacional do Município de Assis, 2012. 47 p.

Orientador: Profº. Esp. Guilherme de Cleva Farto
Trabalho de Conclusão de Curso
Instituto Municipal de Ensino Superior de Assis – IMESA.

1. Segurança. 2.Monitoramento remoto. 3.Google Android. 4.Smartphones.

CDD: 001.6
Biblioteca da FEMA

DEDICATÓRIA

Dedico este trabalho a Deus, responsável pelas forças que me guiaram durante os quatro anos de curso, sem as quais eu não encontraria em mim capacidade de prosseguir.

AGRADECIMENTOS

Em primeiro lugar a Deus, por estar incondicionalmente ao meu lado na realização deste e demais trabalhos no decorrer do curso. Por Sua força e graça, fundamentais para eu prosseguir.

À minha família, por me apoiar dentro de suas possibilidades.

Aos meus amigos por estarem ao meu lado nos bons e maus momentos nos últimos anos.

Ao professor Guilherme de Cleva Farto, pela orientação e apoio na execução deste trabalho.

A todos os professores do curso de Ciência da Computação pelos ensinamentos transferidos ao longo dos anos e que somados compõem o conhecimento indispensável à minha formação. Às professoras Marisa Atsuko Nitto e Regina Fumie Eto pelo apoio no planejamento desse trabalho. .

Aos colegas de turma no curso de graduação cujo companheirismo eu não poderia deixar de agradecer. Em especial ao Eduardo Manarin Daguano, pelo apoio e amizade que foram indispensáveis nos anos iniciais do curso.

Ao Jarley Vaz da Silva, pela amizade, o amor e a força que me ajudou a persistir.

RESUMO

Este trabalho tem por objetivo principal analisar a necessidade de segurança vivida no cenário atual onde é cada vez maior a incidência de crimes contra o patrimônio alheio. É apresentado o projeto de desenvolvimento de um software para monitoramento de ambientes como alternativa à prevenção contra prejuízos patrimoniais.

O motivo para escolha do tema é a certeza de que o aplicativo irá satisfazer as necessidades de um número elevado de usuários. O aplicativo Irá prover possibilidade de acompanhamento em tempo real das ocorrências em determinado local através de câmeras IP munidas com sensor de presença.

As imagens capturadas pelas câmeras serão acessíveis por meio de dispositivos móveis, como *Tablets* e *Smartphones* munidos de sistema Android. A cada alerta de intrusão detectada pelo sensor de presença, o usuário é avisado através de mensagem SMS (Short Message Service). A interface da aplicação é de fácil compreensão, possibilitando navegação entre as telas de forma intuitiva, com ícones e botões auto compreensíveis. Foi utilizada UML (Unified Modeling Language) para padronização dos diagramas e entendimento do aplicativo. Para programação e desenvolvimento foram utilizadas as tecnologias JAVA e Google Android, por serem tecnologias em alta no mercado de TI e juntas propiciarem as melhores soluções para desenvolvimento móvel.

No decorrer do trabalho, foram encontradas dificuldades na recuperação do fluxo de imagens da câmera, assim como no envio de SMS. Ao final, foi obtido êxito na construção do mesmo, e definidas as possibilidades de continuação do aplicativo, como um produto para comercialização.

Palavras – chave: segurança, monitoramento remoto, Google Android, smartphones.

ABSTRACT

This study aims to analyze the main security needs experienced in the current scenario where it is increasing the incidence of crimes against property of others. It presented the project to develop software for monitoring environments as an alternative to prevent property damage.

The reason for choosing the topic is sure that the application will meet the needs of a large number of users. The application will provide possibility for real-time monitoring of events in a given location through IP cameras fitted with motion sensor.

The images captured by the cameras will be accessible through mobile devices such as Smartphones and Tablets equipped with Android system. Each intrusion alert detected by presence sensor, the user is notified via SMS (Short Message Service). The application interface is easy to understand, enabling navigation between screens intuitively, with self understandable icons and buttons. We used UML (Unified Modeling Language) diagrams for standardization of understanding and application. For programming and development technologies were used JAVA and Google Android, as they are in high technologies in the IT market and together they encourage the best solutions for mobile development.

During the work, difficulties were encountered in recovering the flow of images from the camera, as well as sending SMS. In the end, success was obtained in the same building, and defined the possibilities for further application as a product for marketing.

Key - words: security, remote monitoring, Google Android, smartphones.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

Domótica é a área da engenharia e do conhecimento que provê o gerenciamento dos recursos habitacionais. O termo domótica é o resultado da junção da palavra *Domus*, que em latim significa residência, e robótica: área da mecatrônica que utiliza conceitos de programação e eletrônica, voltados ao desenvolvimento de soluções automatizadas. A domótica possui recursos aplicáveis a diversos segmentos. Entre eles, pode-se destacar gestão de energia, comunicação, conforto, entretenimento e segurança.

A segurança é o segmento mais procurado da automação residencial, pois passa por uma transição entre o que é supérfluo e o primordial. Sistemas de segurança inteligentes propõem soluções que vão além da proteção ao patrimônio: podem em muitas situações ser corresponsáveis pela preservação da vida.

Os usuários de tecnologia formam um grupo cada vez maior e mais conscientes em relação aos seus benefícios. Por esse motivo a indústria de TI é altamente propensa à criação de novos sistemas e tecnologias que atendam as expectativas do seu público alvo.

Sistemas de vigilância estão sendo consumidos por um número cada vez maior de usuários, sejam eles residenciais, comerciais ou industriais, pois representam uma maneira eficaz de prover tanto segurança quanto monitoria de pessoas, comportamentos ou processos.

O aplicativo objeto desse trabalho poderá ser utilizado por empresas de segurança, que poderão se responsabilizar por verificar alertas de invasão e enviar autoridades policiais ao ambiente monitorado em caso de confirmação da intrusão. Poderá também ser utilizado por usuários domésticos, para monitoramento da própria residência, ou ainda por organizações empresariais para controle de processos industriais ou controle da produtividade de colaboradores.

1.1 Objetivo

O objetivo desse trabalho é apresentar a proposta de criação de um sistema de segurança por monitoramento remoto. O aplicativo será desenvolvido em Android e composto por câmeras IP, munidas de sensor de presença e movimento, com alertas ao usuário através de mensagens SMS.

A aplicação terá extrema importância para pessoas que passam grande parte do tempo fora de casa, e querem se certificar de que seu patrimônio esteja protegido.

Surtirá efeito também, para empregadores que querem causar em seus funcionários a sensação de serem observados em cem por cento do tempo, com intuito de leva-los a procurar satisfazer melhor as necessidades da empresa.

1.2 Justificativa

Nos dias atuais, segurança é uma preocupação constante. O índice de criminalidade aumenta de forma brusca, tanto nas metrópoles quanto em cidades menores.

Administrar a segurança de bens patrimoniais é um desafio constante. Pessoas estão sempre dispostas a adquirir sistemas de segurança que realmente funcionem, e dê a elas a sensação de tranquilidade.

Ter na palma das mãos o controle sobre o que acontece no ambiente desejado, não importando o lugar do mundo onde o usuário esteja, sem dúvidas é um grande atrativo.

A escolha desse trabalho foi motivada pela certeza de que é um projeto com grandes possibilidades de alcançar um número expressivo e expansível de usuários.

1.3 Público alvo

O público a que se destina a utilização do sistema é composto por empresas de segurança e monitoramento, usuários domésticos e empresas que visem controle de produção e produtividade.

1.4 Estrutura do Trabalho

Este trabalho está dividido em capítulos, estruturados de forma clara e organizada, propiciando ao leitor um claro entendimento dos conceitos abordados, e metodologias utilizadas para sua realização.

O primeiro capítulo apresenta a proposta de desenvolvimento, os objetivos e a justificativa para a criação do aplicativo objeto deste trabalho.

No segundo capítulo é apresentado o embasamento teórico, assim como a descrição das tecnologias utilizadas para o desenvolvimento da aplicação.

O terceiro capítulo contempla o levantamento de requisitos, a especificação do aplicativo e os diagramas obtidos na análise: Diagrama de Casos de Uso, Diagrama de Classes, Diagrama Entidade Relacionamento, Diagrama de Sequências, Diagrama de Atividades.

O quarto capítulo descreve a *Work Breakdown Structure* (WBS), o sequenciamento de atividades e o orçamento do aplicativo.

O quinto capítulo demonstra a implementação do aplicativo, com exposição parcial do código fonte. O capítulo também contém imagens da interface gráfica.

O capítulo seis apresenta as considerações finais sobre o desenvolvimento da aplicação expondo os resultados obtido. Na subseção do mesmo capítulo , são descritas as oportunidades geradas pela construção do aplicativo, tornando claras as possibilidades de continuação do projeto através de trabalhos futuros.

2. REVISÃO DE LITERATURA

Para a construção do sistema são necessárias algumas tecnologias que serão abordadas neste capítulo. A estrutura do sistema conta com câmeras IP com sensores de movimento e presença, posicionadas estrategicamente no ambiente a ser monitorado. É também necessário ter conexão com Internet, um modem 3G, um servidor de vídeo web e um celular com o sistema operacional Google Android. Como é possível verificar, o custo é inferior ao de um Circuito Fechado de TV (CFTV), pois não é necessário ter um computador, visto que as câmeras se comunicam diretamente ao modem.

2.1Câmeras IP

Câmeras IP, ou Camera Internet Protocol são câmeras que se conectam a uma rede via porta Ethernet e são usadas para transmissão de imagens ao vivo pela internet. A tecnologia de monitoramento por IP é uma alternativa aos sistemas de gravação de vídeo *Digital Video Recorder (DVR)*, pois além de prover um sistema de vigilância e monitoramento digital de alto desempenho, agrega o benefício de possuir custo inferior.

Na criação de sistemas de monitoramento, várias câmeras e servidores de vídeos podem ser conectados à rede Ethernet. O protocolo IP, vem sendo usado atualmente para prover a transição analógico-digital dos sistemas de CFTV.

A digitalização do vídeo é feita na própria câmera, ou através do servidor de vídeo, no caso de câmeras analógicas. A utilização dessa tecnologia é altamente viável, pois além do desempenho mais elevado em relação a tecnologias anteriores, o custo é minimizado pelo fato de o sistema dispensar o uso de cabos coaxiais, por fazer conexão direta à porta da rede. Outra vantagem é o fato de as câmeras operarem sem

o uso de computadores, já que possuem seus próprios sistemas de operação e compressores de imagens.

2.2 Google Android

O Android é um sistema operacional open source para celulares, desenvolvido pela Google. Está inclinado a se tornar a plataforma dominante no mercado de dispositivos móveis nos próximos anos.

A vantagem que mais chama atenção dos desenvolvedores é a SDK aberta. Ao contrário de plataformas fechadas como o iPhone, é possível baixar a SDK e estudar o sistema. Outro atrativo é a adição de novos recursos aos celulares, o que aumenta a procura por processadores mais velozes impulsionando o desenvolvimento e venda de novos produtos (Morimoto, 2008, p.6).

No tocante aos sistemas de monitoramento remoto via celular, o Android é a opção mais eficaz e procurada atualmente, pelo fato de ser open source, e permitir a execução de sistemas especificamente desenvolvidos para cada usuário.

A partir da aplicação, o usuário poderá executar ações como adicionar ou remover câmeras, acompanhar as imagens em tempo real, e administrar a gravação de vídeos.

Na condição de administrador, um usuário poderá gerenciar o acesso de outros usuários através de criação de contas autenticadas por *login* e senha.

2.3 UML

UML ou *Unified Modeling Language* é uma linguagem utilizada para modelagem dos dados, ou seja, com ela é possível desenhar através de diagramas as características e atribuições do sistema, permitindo uma visualização prévia do que será feito.

Além da representação gráfica, é também possível representar os significados ou semântica do sistema, facilitando o entendimento do mesmo por parte de quem irá programar.

Através da UML, é possível definir na fase de análise, a especificação, documentação, e estruturação completa de um sistema. Existem no mercado, à disposição de analistas, várias ferramentas que auxiliam nas representações gráficas da UML.

A ferramenta escolhida para a preparação da análise foi o Argo UML, uma ferramenta CASE, open source de modelagem que inclui suporte para todos os diagramas UML.

2.4 Java

Java é uma linguagem de programação orientada a objetos. É uma grande atrativo no mercado de TI atualmente, por sua robustez e simplicidade ao mesmo tempo. A orientação a objetos presente na linguagem permite ao desenvolvedor programar de forma clara, com sintaxe simples. É uma linguagem multitarefa, considerando que vários processos podem executar simultaneamente de maneira concorrente.

A compilação do código ocorre para a JVM (Java Virtual Machine), e é independente da plataforma, por isso é considerada uma linguagem Universal e Interpretada. O nível de segurança oferecido em Java é alto, justificando a grande aceitação no mercado, uma vez que códigos menos passíveis de erro se tornam atraentes em uma era em que, os clientes e usuário de soluções tecnológicas são cada vez mais exigentes quanto ao funcionamento eficaz de uma aplicação. O ponto chave para a otimização dos recursos da linguagem, é a utilização de conceitos de orientação a objetos, pois a linguagem é destinada a esse paradigma.

O ambiente para o desenvolvimento da aplicação objeto desse trabalho é o Eclipse IDE, uma plataforma de código aberto para desenvolvimento de softwares. O envio de mensagens SMS para alertas de intrusão, é feito a através de uma classe

Java, chamada *SmsManager*. O aplicativo verifica o saldo de créditos disponível no cartão SIM do celular, e através dessa classe realiza o envio da mensagem.

3 ANALISE E ESPECIFICAÇÃO

Este capítulo apresenta informações obtidas a partir da análise de requisitos do Aplicativo.

3.1 Diagrama de Casos de Uso

O diagrama de casos de uso especifica o comportamento do sistema. O diagrama descreve o cenário e a sequência das ações desempenhadas.

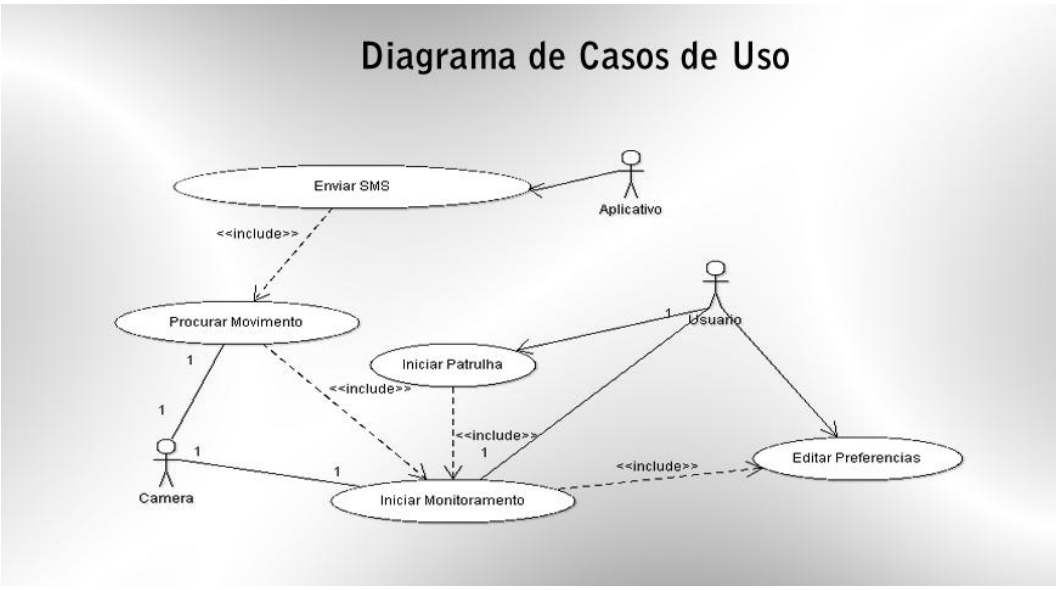

Figura 1 Diagrama de Casos de Uso

Lista de Casos de uso

1. Editar Preferências
2. Iniciar Monitoramento
3. Iniciar Patrulha
4. Procurar Movimento
5. Enviar SMS

3.2 Especificação de Casos de Uso

UC_1 – Editar Preferências

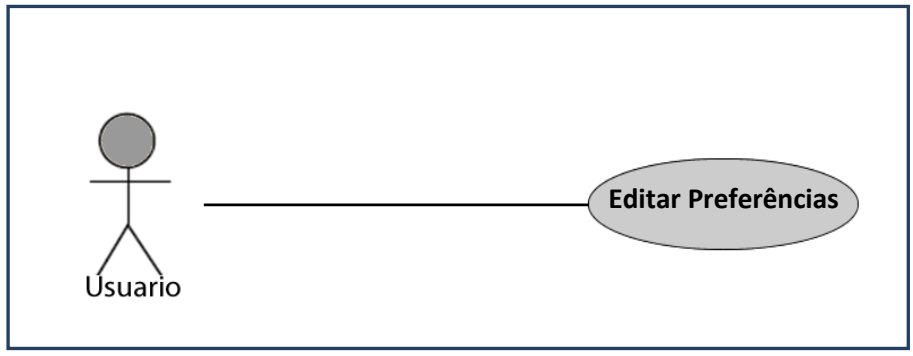

Figura 2 UC_1 Editar Preferências

CASO DE USO UC_01: UC – Editar Preferências

1. Finalidade/Objetivo
   - Selecionar a câmera sobre a qual se deseja visualizar Imagens. A seleção ocorre por meio da digitação do Host, que é o IP da câmera desejada e da Porta liberada para acesso no modem. Além dessas informações, o usuário deverá se conectar à câmera por meio de *login* e senha. O *login* e a senha são especificados via browser no aplicativo disponibilizado pela câmera. Nessa tela também são editados o número de telefone destino para envio de SMS caso haja detecção de movimento. O usuário também digita o mensagem padrão a ser enviada.

2. Atores
   - Usuário.

3. Pré-Condições
   - Estar na janela de *"Preferências"*.

4. Evento Inicial
   - O usuário deve selecionar no *Menu Principal* do Sistema a opção *Preferências*.

5. Fluxo Principal

a) O Usuário seleciona a opção no *MenuPrincipal*.
b) O caso de uso é iniciado.
c) O usuário digita as informações necessárias.
d) O usuário salva as informações
e) O usuário retorna ao Menu Principal.

6. Fluxos Alternativos

A1-Cancela a Operação.
a) O usuário cancela o caso de uso sem editar as preferências. O Sistema retorna ao *Menu Principal*.

7. Fluxos de Exceção

E1- O usuário digita informações de host, porta ou autenticação incorretas.
a) O sistema realiza um teste, verifica que as informações não são condizentes com as configurações do dispositivo e nega o acesso.

8. Pós Condições

- O usuário poderá visualizar o ambiente e controlar os movimentos da câmera.

9. Casos de Testes

- Verificar o domínio dos campos.
- Cancelar o caso de uso.

UC_2 – Iniciar Monitoramento

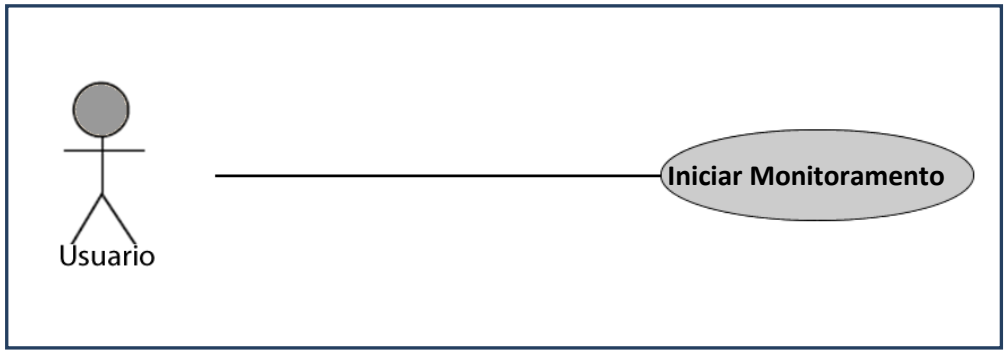

Figura 3 UC_2 Iniciar Monitoramento

CASO DE USO UC_02: UC – Iniciar Monitoramento

1. Finalidade/Objetivo
- Visualizar as imagens da câmera através do aplicativo.

2. Atores
- Usuário.

3. Pré-Condições
- Estar no Menu Principal.

4. Evento Inicial
- O usuário deve selecionar no *Menu Principal* do Sistema a opção *Iniciar Monitoramento*.

5. Fluxo Principal
a) O Usuário seleciona a opção no *MenuPrincipal*.
b) O caso de uso é iniciado.
c) O usuário visualiza o ambiente monitorado pela câmera.

6. Fluxos Alternativos
A1-Iniciar Patrulha vertical ou Iniciar Patrulha horizontal.
a) O usuário pode controlar os movimentos da câmera através de setas direcionais ou clicando nos botões *Patrulha Vertical* e *Patrulha Horizontal*.

.

A2-Cancela a Operação.
a) O usuário retorna ao Menu Principal.
b) O usuário pode então realizar outras operações.

7. Pós Condições
- Caso o sensor da câmera detecte algum movimento, o aplicativo envia SMS ao número salvo nas preferências .

8. Casos de Testes
- A câmera verifica constantemente se há movimentação no local monitorado.

UC_3 – Baixar Gravação

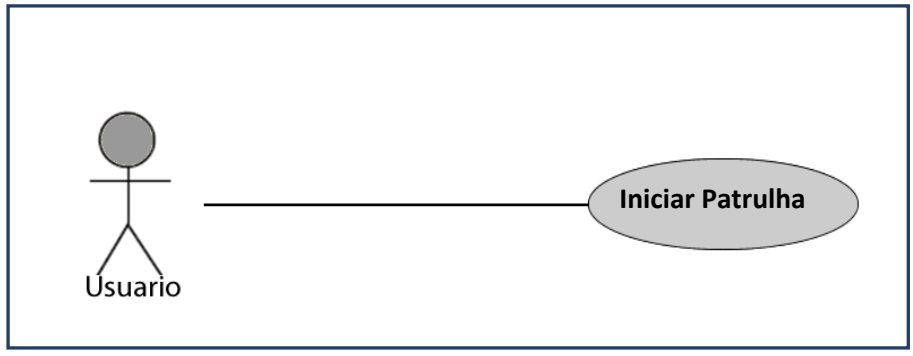

Figura 4 UC_3 Iniciar Patrulha

CASO DE USO UC_03: UC – Iniciar Patrulha

1. Finalidade/Objetivo
- Controlar a direção dos movimentos da câmera.

2. Atores
- Usuário.

3. Pré-Condições
- Estar na janela de *"Monitoramento"*.

4. Evento Inicial
- O usuário deve clicar em um dos botões, *"Patrulha Horizontal"* ou *"Patrulha Vertical"*.

5. Fluxo Principal
   a) O Usuário acessa a janela *"Arquivos de Monitoramento"*.
   b) O caso de uso é iniciado.
   c) O usuário clica sobre o botão desejado.
   d) A câmera se movimenta respondendo aos comandos do botão.

6. Fluxos Alternativos

A1-Cancela a Operação.
   a) O usuário cancela o caso de uso sem realizar nenhuma patrulha.

7. Pós Condições
- Caso haja movimentação no local monitorado, o sistema envia SMS ao número definido nas preferências.

8. Casos de Testes
- Verificar o movimentação no ambiente.
- Cancelar o caso de uso.

UC_4 – Procurar Movimento

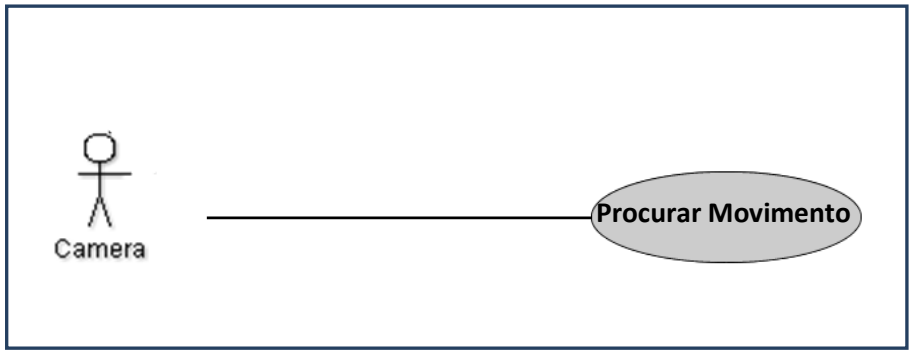

Figura 5 UC_4 procurar Movimento

CASO DE USO UC_04: UC – Procurar Movimento

1 .Finalidade/Objetivo
- Procurar por movimentações no ambiente que possam representar intrusão.

2 Atores
- Câmera.

3 Pré-Condições
- Estar na janela de *"Monitoramento "*.

4 Evento Inicial
- Ter cadastrado as informações da câmera ao iniciar o aplicativo.

5 Fluxo Principal

a) O Usuário digita as preferências da câmera e inicia o monitoramento.
b) A câmera passa a patrulhar o ambiente em busca dos movimentos.

6 Pós Condições

- Caso haja movimentação no local monitorado, o sistema envia sms ao número definido nas preferências.

7 Casos de Testes

- Verificar o movimentação no ambiente.
- Cancelar o caso de uso.

UC_5 – Enviar SMS

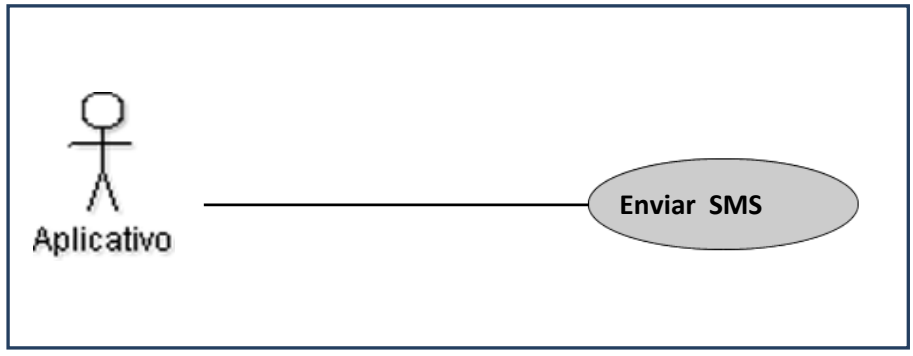

Figura 6 UC_5 Enviar SMS

CASO DE USO UC_05: UC – Enviar SMS

1. Finalidade/Objetivo

- Enviar mensagem de alerta em caso de intrusão no ambiente monitorado.

2. Atores

- Aplicativo.

3 Pré-Condições

- A câmera ter detectado o movimento através do sensor de presença.

4 Evento Inicial

- A câmera deve estar executando a patrulha e monitoramento.

3 Fluxo Principal

a) O Usuário salva as preferências da câmera.
b) A câmera inicia o monitoramento.
c) O sensor de presença da câmera detecta o movimento.
d) O aplicativo envia o SMS ao número cadastrado nas preferências, utilizando o crédito no cartão SIM do dispositivo móvel.

4 Casos de Testes

- Verificar o número informado pelo usuário em Preferências.
- Verificar se há saldo disponível no cartão SIM.

3.3 Diagrama de Classes

O diagrama de classes permite visualização estática do projeto para posterior especificação e documentação do modelo estrutural.

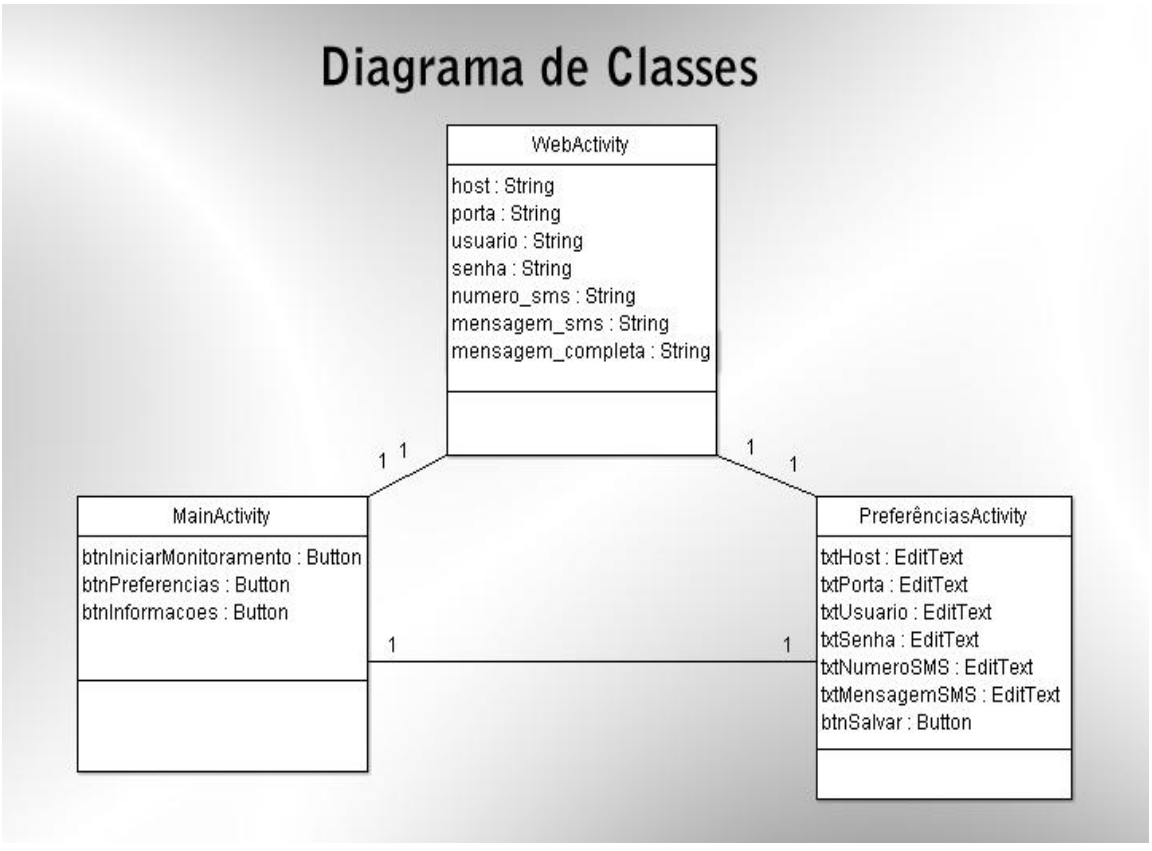

Figura 7 – Diagrama de Classes

3.4 Diagrama de Atividades

Os diagramas de atividade são um recurso essencial da Unified Modeling Language (UML), pois através dele se pode obter o fluxo de controle de uma atividade para outra.

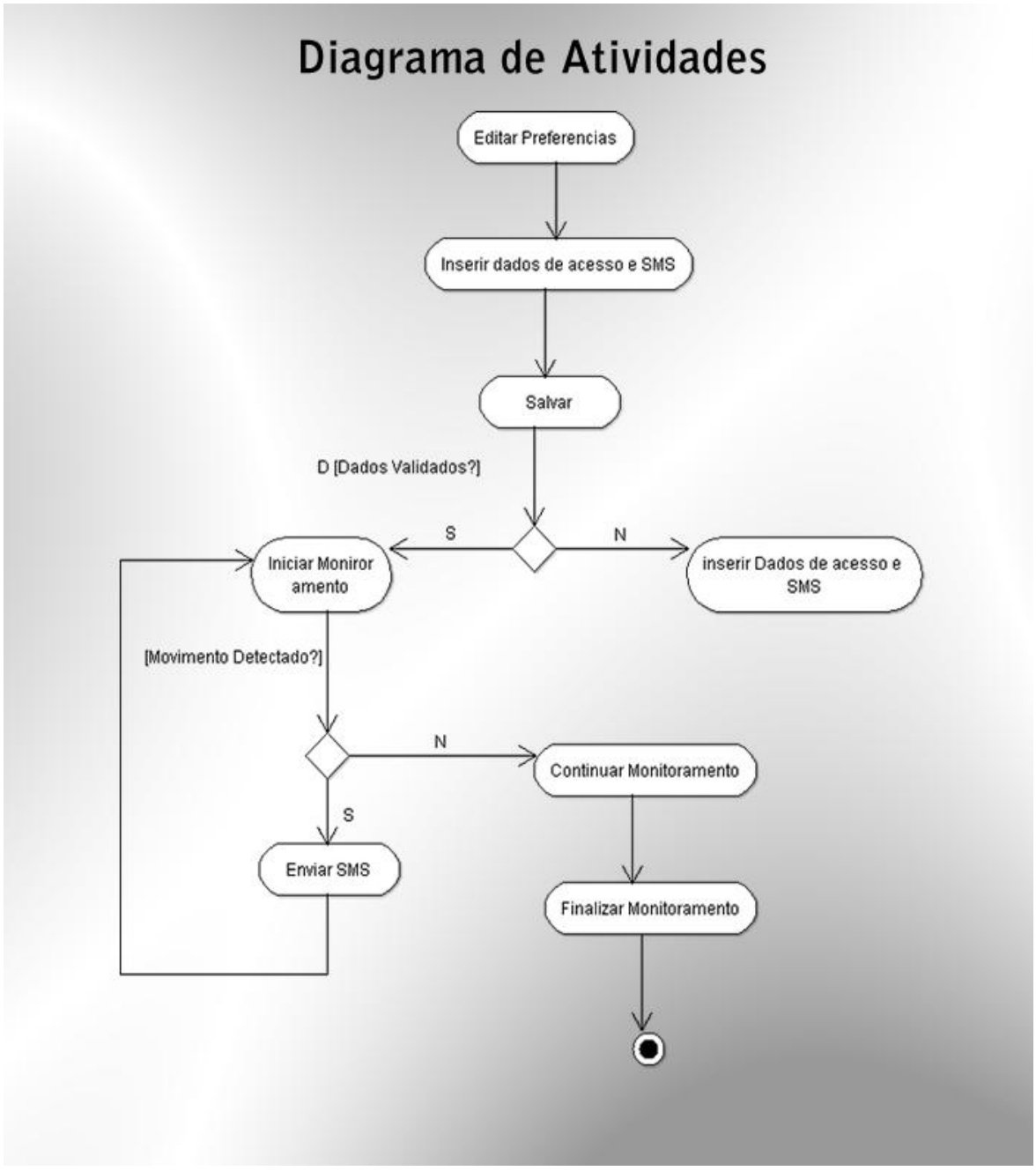

Figura 8 Diagrama de Atividades

4 ESTRUTURA DO PROJETO

Este capítulo apresenta a metodologia de desenvolvimento do trabalho. As fases e etapas de sua construção serão ilustradas no diagrama da Estrutura Analítica do Projeto, também conhecida como WBS (Work Breakdown Structure). O atual capítulo apresenta também o diagrama de sequência de atividades e orçamento do sistema.

4.1 Estrutura Analítica do Projeto

Através da Estrutura Analítica do Projeto é possível dividir o trabalho em subpartes, visando o melhoramento da distribuição das tarefas associadas e facilitar o gerenciamento do projeto. É possível obter uma visão ampla, facilitando a divisão do conjunto de atividades a serem executadas em cada fase.

Para a realização deste projeto, as tarefas foram divididas conforme ilustrado na figura 9, visando a obtenção dos resultados esperados.

ESTRUTURA ANALÍTICA DO PROJETO

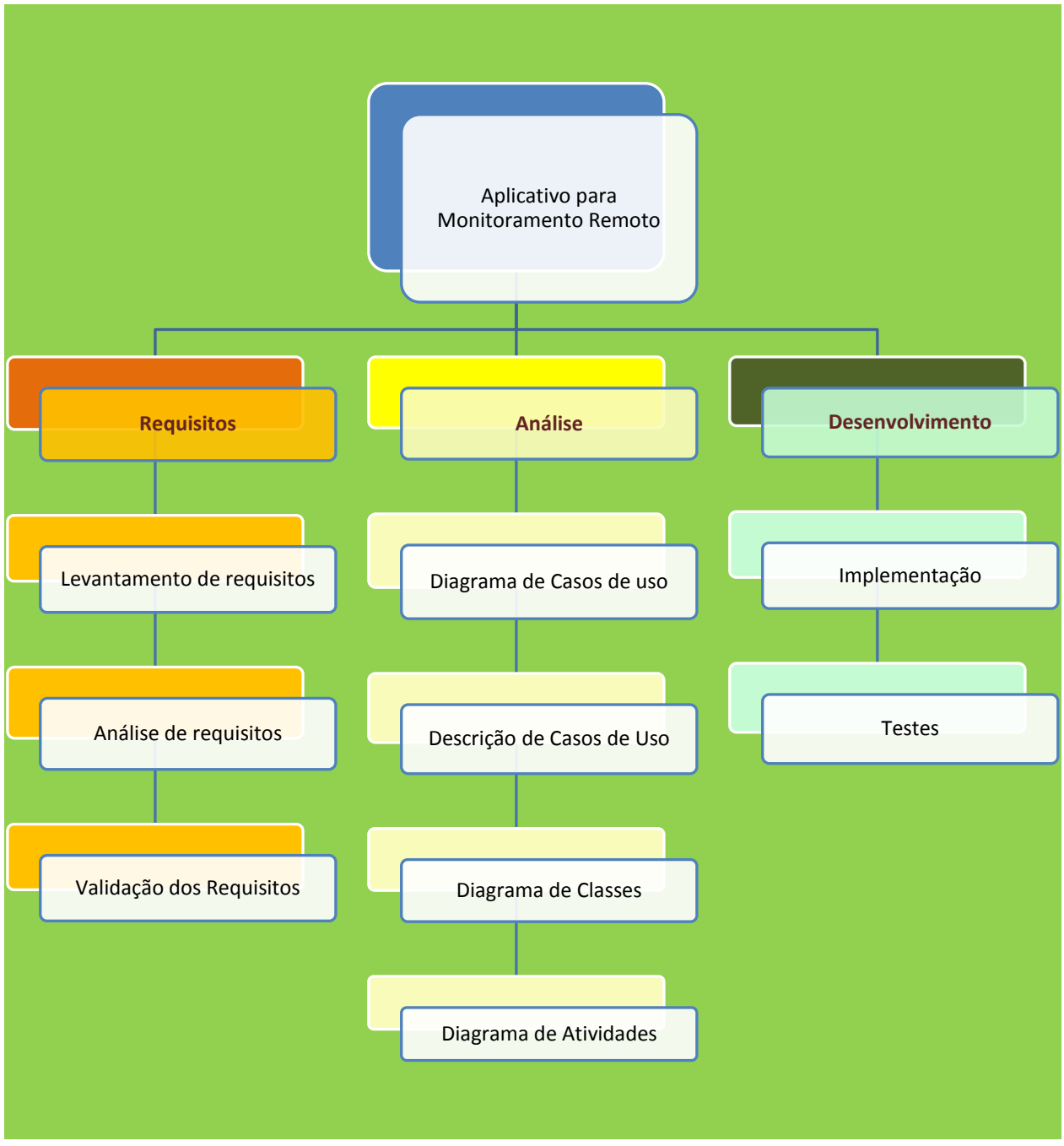

Figura 9 Work Breakdown Structure

4.2 Sequenciamento das Atividades

No diagrama de Sequenciamento de Atividades é apresentado o tempo de duração de cada atividade desenvolvida ao longo do projeto.

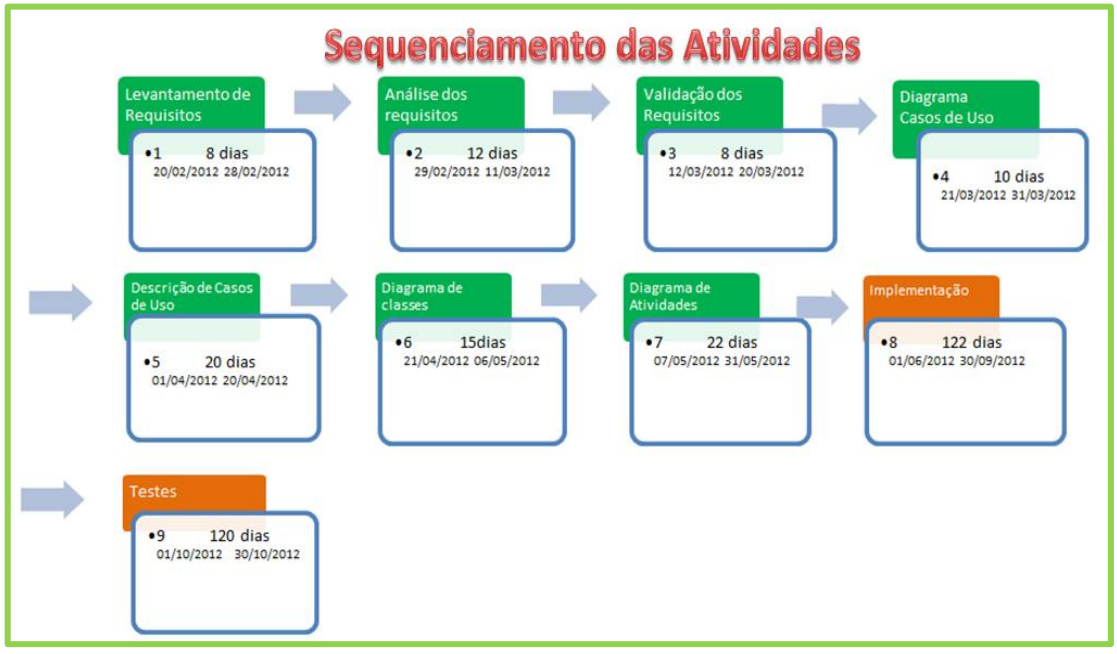

Figura 10 Sequenciamento das Atividades

4.3 Orçamento

Os recursos necessário para a análise e desenvolvimento do aplicativo objeto desse trabalho são:

- 01 Analista de Sistema
- 01 Programador
- 01 Notebook
- 01 Câmera IP

4.3.1 Análise e Programação

Os custos com análise e programação ocorreram de acordo com a figura abaixo, onde cada uma das 115 horas investidas em análise custou R$ 35,00, totalizando o valor de R$ 4.025,00. Em programação, foram investidas 100 horas, com valor unitário de R$ 28,00, compondo um total de R$ 2.800,00. O gasto total em análise e programação foi de R$ 6.825,00.

Analista de Sistemas			
Analista	Quantidade de Horas	Valor da Hora	Total
Roberta F. de Souza	115	R$ 35,00	R$ 4.025,00
Programador			
Analista	Quantidade de Horas	Valor da Hora	Total
Roberta F. de Souza	100	R$ 28,00	R$ 2.800,00
		TOTAL	R$ 6.825,00

Figura 11 Custo Analise e Programação

4.3.2 Equipamentos

Os recursos físicos necessários para o desenvolvimento do aplicativo foram um Notebook Dell, no valor de R$ 2.200,00 com depreciação diária de R$ 3,05, sendo que o gasto total do equipamento corresponde ao valor de R$ 727,00. Fez-se também necessária a compra de uma câmera IP no valor de R$ 290,00, com depreciação diária de R$ 0,39, totalizando o valor de R$ 98,50. O total geral gasto em equipamentos é de R$ 825,50.

Equipamentos					
Equipamento	Valor		Depreciação Diária	Total	
Notebook Dell	R$	2.200,00	R$ 3,05	R$	727,00
Camera IP Easyn	R$	290,00	R$ 0,39	R$	98,50
		TOTAL		R$	825,50

Figura 12 Custo Equipamentos

4.3.3 Custo total do Projeto

O custo total para o desenvolvimento do aplicativo é de R$ 7.650,50, considerando-se a somatória de gastos com mão de obra e equipamentos.

Valor Total do Projeto		
Mão de Obra	R$	6.825,00
Equipamento	R$	825,50
TOTAL	R$	7.650,50

Figura 13 Custo Total do Projeto

5 IMPLEMENTAÇÃO

Para a implementação do aplicativo, foi utilizado o ambiente de desenvolvimento *Eclipse IDE.* A aplicação é baseada na plataforma Google Android para dispositivos móveis, que fornece ferramentas para criação de aplicativos e utiliza a linguagem Java.

5.1 Organização do Projeto

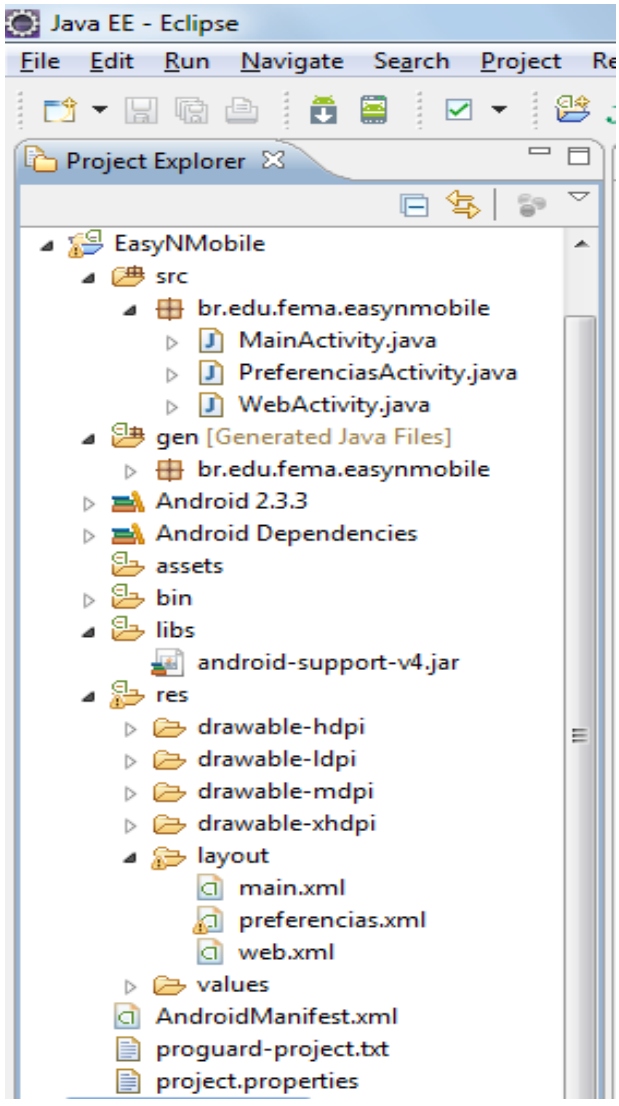

Figura 14 Organização do Projeto

O projeto está dividido em três classes Java: *MainActivity, PreferenciasActivity* e *WebActivity*. As classes são organizadas no pacote **br.edu.fema.easynmobile**.

MainActivity – É a classe principal, que gerencia o acesso às telas do sistema.

PreferenciasActivity - É a classe que gerencia as informações de autenticação de usuário, e informações de rede para acesso à câmera.

WebActivity - É a classe que recupera da *web* as imagens da câmera, e gerencia as movimentações e detecção e movimento. Essa classe também contém o método responsável por enviar SMS com notificação de Intrusão.

5.2 Classes

5.2.1 Classe MainActivity

A classe *MainActivity* apresenta o método *onCreate()* conforme figura a seguir.

O método *onCreate( )* é o método onde é feita a inicialização dos componentes essenciais de uma atividade. Dentro dele, o *layout* é invocado em *setContentView( )*. Ainda dentro de *onCreate( )* é criada uma *intent* para inicializar cada atividade (*activity)*.

Segue abaixo, o Método *onCreate( )* da classe MainActivity. O método é responsável por inicializar os componentes essenciais de uma atividade. Nesse caso, são inicializados e tratados os notões *Iniciar Monitoramento, Preferências e Informações*.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    btnIniciarMonitoramento = (Button) findViewById(R.id.btnIniciarMonitoramento);
    btnPreferencias = (Button) findViewById(R.id.btnPreferencias);
    btnInformacoes = (Button) findViewById(R.id.btnInformacoes);

    btnIniciarMonitoramento.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View arg0) {
            Intent iWebActivity = new Intent(MainActivity.this, WebActivity.class);
            startActivity(iWebActivity);
        }
    });
```

```
    btnPreferencias.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View arg0) {
            Intent iPreferenciasActivity = new Intent(MainActivity.this, PreferenciasActivity.class);
            startActivity(iPreferenciasActivity);
        }
    });

    btnInformacoes.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View arg0) {
             Intent iInformacoesActivity = new Intent(MainActivity.this, InformacoesActivity.class);
             startActivity(iInformacoesActivity);
        }
    });
}
```

5.2.2 Classe PreferenciasActivity

A classe *PreferenciasActivity* é responsável pela manipulação dos dados de acesso e preferências do aplicativo.

Possui os métodos *onCreate*, *carregarPreferencias* e *salvarPreferencias.*

No método *onCreate( )* são capturadas as preferências digitadas pelo usuário. Essas preferências são salvas em *setOnClickListener( )*, onde é invocado o método *salvarPreferencias( )* que por sua vez, realiza a persistência dos dados através de *SharedPreferences*, uma classe que fornece um framework geral, capaz de salvar e recuperar dados de tipos primitivos.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.preferencias);

    txtHost = (EditText) findViewById(R.id.txtHost);
    txtPorta = (EditText) findViewById(R.id.txtPorta);
    txtUsuario = (EditText) findViewById(R.id.txtUsuario);
    txtSenha = (EditText) findViewById(R.id.txtSenha);
    txtNumeroSMS = (EditText) findViewById(R.id.txtNumeroSMS);
    txtMensagemSMS = (EditText) findViewById(R.id.txtMensagemSMS);
    btnSalvar = (Button) findViewById(R.id.btnSalvar);
```

```
        btnSalvar.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
                try {
                    salvarPreferencias();

                    Toast.makeText(PreferenciasActivity.this, "Preferências salvas com sucesso.", Toast.LENGTH_LONG).show();
                } catch (Exception e) {
                    Toast.makeText(PreferenciasActivity.this, "Falha ao salvar preferências.\n\nErro: " + e.getMessage(), Toast.LENGTH_LONG)
                }
            }
        });

        try {
            carregarPreferencias();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

O método *carregarPreferencias( ),* carrega na tela os dados padrão, informados via programação. O usuário pode substituir esses dados pelas informações de acesso e login da câmera desejada.

```
    public void carregarPreferencias() throws Exception {
        SharedPreferences prefs = getSharedPreferences("CAMERA_IP", Activity.MODE_PRIVATE);

        String host = prefs.getString("HOST", "192.168.1.9");
        String porta = prefs.getString("PORTA", "27");
        String usuario = prefs.getString("USUARIO", "roberta");
        String senha = prefs.getString("SENHA", "admin");
        String numeroSMS = prefs.getString("NUMERO_SMS", null);
        String mensagemSMS = prefs.getString("MENSAGEM_SMS", null);

        txtHost.setText(host);
        txtPorta.setText(porta);
        txtUsuario.setText(usuario);
        txtSenha.setText(senha);
        txtNumeroSMS.setText(numeroSMS);
        txtMensagemSMS.setText(mensagemSMS);
    }
```

O método *salvarPreferencias( ),*salva os dados informados pelo usuário através de SharedPreferences, que por sua vez é uma classe que realiza a persistência dos dados, sem o uso de Banco de Dados. Os dados são salvos em um arquivo, nesse caso chamado de “CAMERA_IP”, para posterior consulta, ou seja, mesmo que a aplicação seja morta o arquivo continua a existir.

```
public void salvarPreferencias() throws Exception {
    SharedPreferences prefs = getSharedPreferences("CAMERA_IP", Activity.MODE_PRIVATE);

    SharedPreferences.Editor editor = prefs.edit();

    String host = txtHost.getText().toString();
    String porta = txtPorta.getText().toString();
    String usuario = txtUsuario.getText().toString();
    String senha = txtSenha.getText().toString();
    String numeroSMS = txtNumeroSMS.getText().toString();
    String mensagemSMS = txtMensagemSMS.getText().toString();

    editor.putString("HOST", host);
    editor.putString("PORTA", porta);
    editor.putString("USUARIO", usuario);
    editor.putString("SENHA", senha);
    editor.putString("NUMERO_SMS", numeroSMS);
    editor.putString("MENSAGEM_SMS", mensagemSMS);

    editor.commit();
}
```

5.2.3 Classe WebActivity

A classe WebActivity contém os métodos de controle e manipulação da câmera. Alguns desses métodos são demonstrados a seguir.

Através do método carregarPreferencias( ) da classe webActivity as preferências são carregadas e utilizadas para a conexão entre o aplicativo e as imagens da câmera disponíveis na web, através no IP, armazenado na variável *HOST*.

```
public void carregarPreferencias() throws Exception {
    SharedPreferences prefs = getSharedPreferences("CAMERA_IP", Activity.MODE_PRIVATE);

    HOST = prefs.getString("HOST", "192.168.1.9");
    PORTA = prefs.getString("PORTA", "27");
    USUARIO = prefs.getString("USUARIO", "roberta");
    SENHA = prefs.getString("SENHA", "admin");
    NUMERO_SMS = prefs.getString("NUMERO_SMS", null);
    MENSAGEM_SMS = prefs.getString("MENSAGEM_SMS", null);
}
```

Uma vez carregadas as imagens da câmera, o aplicativo inicia a busca por movimentações no ambiente monitorado. O método *detectarMovimento( )* da classe *webActivity* acessa a URL da câmera, e através de *get_status* obtém informações atualizadas do sensor de presença do dispositivo.

```
private boolean detectarMovimento() throws Exception {
    StringBuilder content = new StringBuilder("");

    URL url = new URL("http://" + HOST + ":" + PORTA + "/get_status.cgi?user=" + USUARIO + "&pwd=" + SENHA);

    BufferedReader in = new BufferedReader(new InputStreamReader(url.openStream()));

    String inputLine = null;

    while ((inputLine = in.readLine()) != null) {
        content.append(inputLine).append("\n");
    }

    in.close();

    return content.toString().contains("var alarm_status=1");
}
```

Caso o movimento seja detectado, através do método *enviarSMSNotificacao( )*, o aplicativo envia um SMS para o número destino informado em *Preferencias*. O método testa se foi informada uma mensagem um número de celular válido. A classe *smsManager* é reponsável por acessar o cartão SIM do celular e a partir dele enviar o SMS.

```
public boolean enviarSMSNotificacao() throws Exception {
    MENSAGEM_COMPLETA = (MENSAGEM_SMS != null ? MENSAGEM_SMS + " - " : "") + "Data: " + customFormater.format(new Date()) + " - Camera: "

    if (NUMERO_SMS != null && MENSAGEM_COMPLETA != null) {
        SmsManager smsManager = SmsManager.getDefault();

        smsManager.sendTextMessage(NUMERO_SMS, null, MENSAGEM_COMPLETA, null, null);

        Toast.makeText(WebActivity.this, "SMS de notificação enviado com sucesso.", Toast.LENGTH_LONG).show();
    } else {
        Toast.makeText(WebActivity.this, "Número ou mensagem de notificação não configurados.\nPor favor, verifique suas preferências.",

        return false;
    }

    return true;
}
```

5.3 Interface do Aplicativo

Ao acessar o sistema, o usuário terá acesso à tela de boas vindas, de onde poderá acionar o *menu* e acessar a tela "*Menu Principal*".

Figura 15 Tela inicial do Aplicativo

No Menu Principal, o usuário poderá clicar em *Informações*, onde obterá informações básicas do aplicativo. Clicando em *Preferências* o usuário pode informar as opções de acesso à câmera e destinatário para envio de SMS. Clicando no botão *Iniciar Monitoramento* é possível finalmente visualizar as imagens da câmera, desde que as preferências tenham sido informadas corretamente.

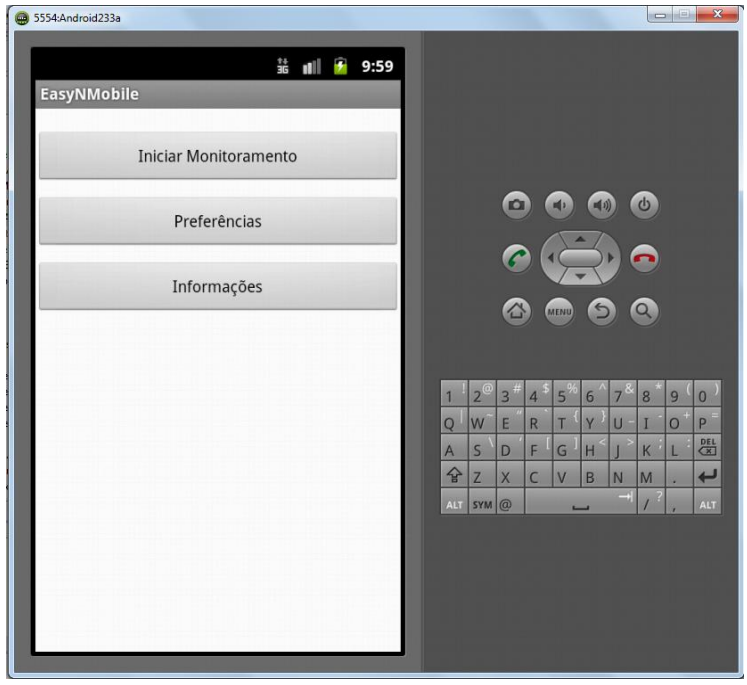

Figura 16 Menu Principal do Aplicativo

Tela Informações, onde é possível obter informações comerciais sobre as funcionalidades do aplicativo.

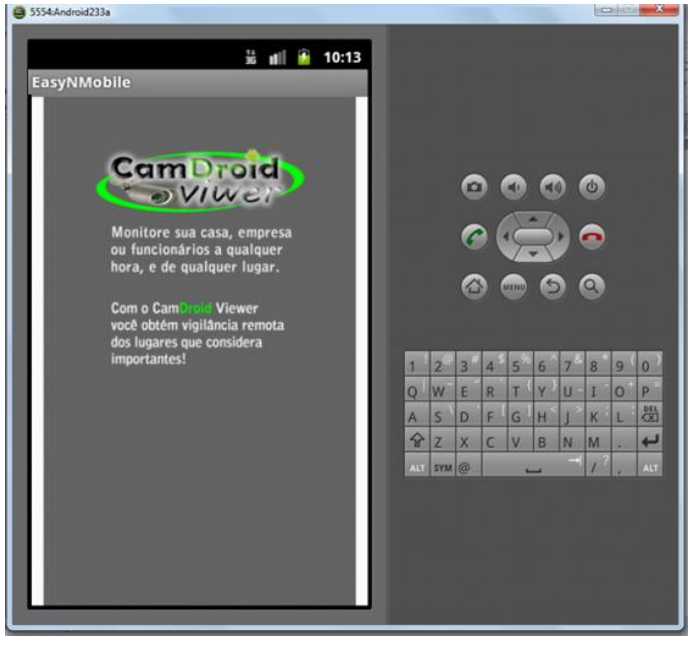

Figura 17 Tela Informações do Aplicativo

Tela Preferências, onde é possível digitar e salvar as informações de acesso à câmera, como IP, porta, usuário, senha, número de celular para envio de SMS em suspeita de invasão e texto do SMS a ser enviado.

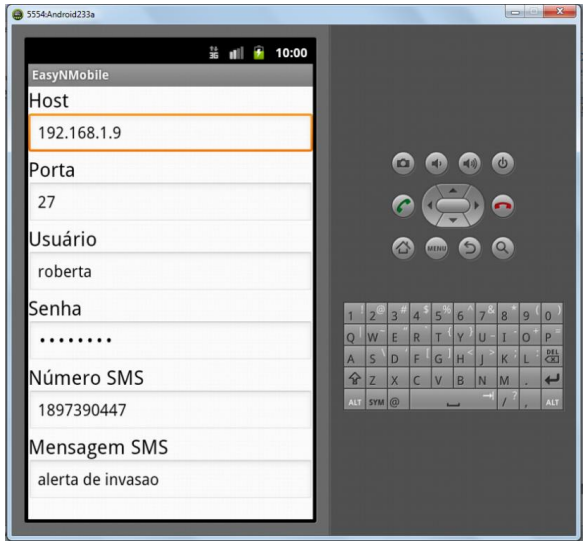

Figura 18a Tela Preferencias do Aplicativo

Ao rolar a barra e clicar no botão *Salvar*, o aplicativo salva as informações e realiza as verificações das informações de acesso.

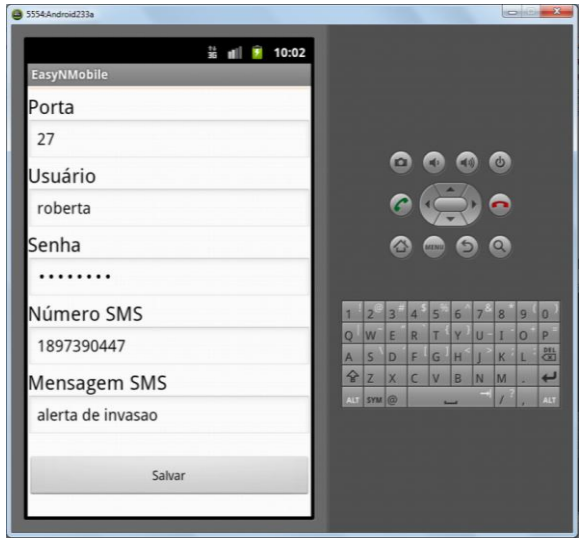

Figura 18b Tela Preferencias do Aplicativo

Ao clicar no botão *Iniciar Monitoramento*, é possível acessar as imagens capturadas pela câmera e controlar seus movimentos através das setas direcionais.

Clicando em *Patrulha Horizontal* ou *Patrulha Vertical*, é possível acionar os movimentos contínuos da câmera. Caso a câmera detecte movimento, o aplicativo envia SMS ao número informado em *Preferências*.

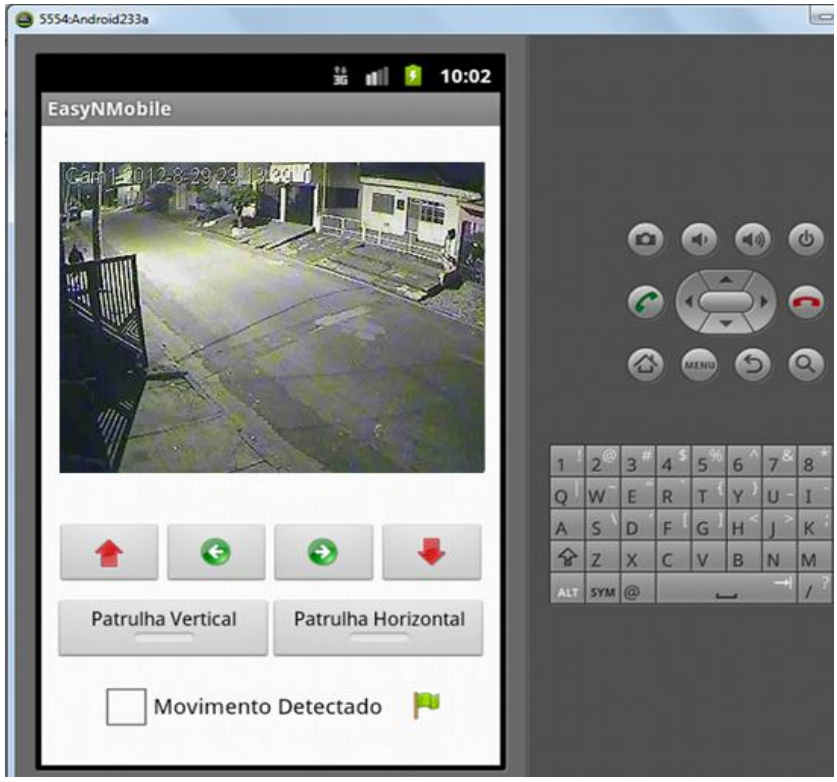

Figura 19 Tela Monitoramento do Aplicativo

6 CONCLUSÃO

A segurança é um segmento em constante ascensão no mercado. Pessoas estão cada vez mais dispostas a investir em soluções que visem sua proteção. O uso da tecnologia a favor desse segmento aumenta a cada dia.

O aplicativo construído nesse trabalho foi desenvolvido para monitoramento remoto de ambientes, proporcionando ao usuário o conforto de se sentir seguro. O usuário poderá por exemplo, visualizar sua casa antes de adentrá-la ao voltar do trabalho, ou ainda, ter controle sobre o que acontece quando está viajando. O alerta via SMS traz o benefício de manter o usuário informado sobre movimentações estranhas.

Durante o desenvolvimento do aplicativo, houve dificuldades com a recuperação do fluxo de imagens da câmera, assim como em criar uma solução para envio de SMS.

6.1 Trabalhos Futuros

Para trabalhos futuros pretende-se aumentar o desempenho de gerenciamento das informações. Os vídeos serão gravados a cada período preestabelecido de tempo para posterior consulta e *download* feitos pelo usuário, sendo armazenados em banco de dados. As mensagens de alerta via SMS, serão disparadas também em decorrência de acionamento de botão do pânico, pois na presença de usuários no ambiente monitorado, o sensor de presença não deve estar ativo.

REFERÊNCIAS

Android Architecture. <http://code.google.com/android/what-is-android.html> Acessado em 28 mar. 2012.

ECLIPSE. **Eclipse - An open development platform**. Disponível em: <http://www.eclipse.org>.Acesso em 10 mar. 2012.

ENGEL, A. P.; CAVALOTTI, G. **Movimentação de Cameras** – Engenharia Elétrica, Telecomunicações PUCPR. Curitiba, 2005. Projeto de Integração. Disponível em: <http://www.ppgia.pucpr.br/~santin/ee/2005/2s/4>. Acesso em 15 abr 2012.

FERREIRA, G. F.; PURGER NT, H.; BUENO JR, M. S. L. **Câmera Pan-Tilt**. Engenharia Elétrica, Telecomunicações PUCPR. Curitiba, 2005. Projeto de Integrração. Disponível em: <http://www.ppgia.pucpr.br/~santin/ee/2005/1s/5>. Acesso em 15 abr. 2012.

Guia do Hardware. **Sistema de Vigilância Digital com ZoneMinder**. Disponível em: <http://www.guiadohardware.net/tutoriais/sistema-vigilancia-zoneminder>. Acesso em 5 abr 2012.

Java Magazine - **Modelagem com ArgoUML**. Disponível em <http://www.devmedia.com.br/artigo-java-magazine-43-modelagem-com-argouml/8627>. Acesso em 9 abr. 2012.

MORIMOTO, CARLOS E. **Entendendo o Google Android**. Disponível em <http://www.hardware.com.br/artigos/google-android/>. Acesso em 10 abr. 2012.

OPEN HANDSET ALLIANCE. **Android operating system**. 2008. Disponível em: <http://www.android.com> Acesso em 10 abr. 2012.

QUINDERÉ, PATRICK ROMERO FROTA. **Casa Inteligente** – Um Protótipo de Sistema de Automação Residencial de Baixo Custo. 2009. 69p. Monografia (Trabalho de Conclusão de curso) – FACULDADE FARIAS BRITO - CIÊNCIA DA COMPUTAÇÃO, Fortaleza 2009

ZONEMINDER. **Linux video camera and cctv security with motion detection**. 2004. Disponível em: <http://www.zoneminder.com>. Acesso em 28 mar. 2012.

ANEXOS

CRONOGRAMA PARA A ESTRUTURA DE DESENVOLVIMENTO

A seguir, é apresentado o cronograma utilizado para a execução deste trabalho.

CRONOGRAMA	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov
Levantamento de Requisitos										
Analise dos Requisitos										
Validação dos Requisitos										
Abstração de Casos de Uso										
Especificação de Casos de Uso										
Diagrama de Classes										
Diagrama de Atividades										
Elaboração da Qualificação										
Exame de Qualificação										
Programação										
Testes										
Escopo da Versão Final										
Apresentação Final - Banca										