MAGREGISTER 1.0: GERADOR DE INTERFACES DE COLETAS DE DADOS PARA PDA'S

Acadêmico: Gilson Chequeto

Orientador: Adilson Vahldick

www.furb.br

Roteiro

- Introdução
- Objetivos do trabalho
- Fundamentação teórica
- Desenvolvimento do trabalho
- Conclusão
- Extensões

Introdução

- Importância do desenvolvimento de aplicativos direcionados para dispositivos móveis
- Necessidade de ferramentas para coletas de dados para o mercado corporativo
- Geração de interfaces para coleta de dados em ambientes externos ao ambiente de um sistema de gestão empresarial
- Ferramenta para a geração de telas para execução em dispositivos móveis

Objetivos do trabalho

- Desenvolver uma ferramenta que permita a geração de interfaces para dispositivos móveis
  - Disponibilizar um modelador de interfaces na forma de uma aplicação desktop, que gravará a modelagem dos leiautes no formato XML
  - Disponibilizar um gerador de código que siga a especificação JME
  - Permitir que os dados sejam coletados e gravados no banco de dados DB2 Everyplace
  - Permitir que os dados coletados no PDA sejam transferidos para PCs através do uso da linguagem XML

Fundamentação teórica

- Conceitos
  - ✓ JME
  - ✓ IBM J9 VM
  - ✓ Geradores de código
  - ✓ *Templates*
  - ✓ Motor de *templates* Velocity
  - ✓ DB2 Everyplace

- Trabalhos correlatos
  - ✓ Aplicativo para representante comercial em dispositivo móvel (PDA) usando a tecnologia JME e banco de dados
  - ✓ Sistema de gerenciamento customizável baseado em PDA's
  - ✓ Protótipo de software para dispositivos móveis utilizando Java ME para cálculo de regularidade em rally

JME

- Vertente da linguagem Java destinada a dispositivos com recursos limitados de memória, vídeo e processamento

- Satisfaz necessidades de:
  - ✓ Consumidores e fabricantes de equipamentos eletrônicos computadorizados
  - ✓ Provedores de serviços que desejam distribuir o conteúdo de suas soluções por meio de dispositivos móveis
  - ✓ Desenvolvedores de sistemas pequenos e limitados

- Possui um conjunto de tecnologias que podem ser usadas na construção de aplicativos:
  - JVMs
  - Bibliotecas especializadas para cada tipo de dispositivo
  - Ferramentas para o desenvolvimento e implantação de softwares e configuração de dispositivos

IBM J9 VM

- Implementa uma arquitetura configurável e compacta
- Provê uma interface comum para aplicações executarem em diferentes dispositivos e sistemas operacionais. Executa nos sistemas operacionais (PalmOS, PocketPC, QNX, Linux embarcado, OSE, ITRON, etc.)
- Gerencia as interfaces específicas com o sistema operacional e com o hardware do dispositivo
- Suporta às configurações CLDC 1.1 e MIDP 2.0
- Utilizada pelo fato de que as máquinas virtuais da Sun não suporta estas configurações

Geradores de código

- Geração de código é a técnica pela qual se constrói código utilizando programas.
- Auxiliam no processo de desenvolvimento de software
- Podem gerar um sistema completo ou somente rotinas específicas
- Etapas para o desenvolvimento de um gerador de código
  - ✓ Identificação da saída
  - ✓ Definição da entrada e sua análise
  - ✓ Interpretação da entrada e formatação da saída
  - ✓ Geração da saída a partir das informações de entrada

Templates

- São arquivos utilizados para a geração de outros arquivos
- Utilizados em geração de código para permitir a padronização do código gerado
- Permitem que a formatação do código gerado esteja externa ao código da aplicação que o gera
- São formados por códigos estáticos e códigos dinâmicos

Motor de *Templates* Velocity

- Não é uma aplicação, mas um conjunto de classes Java
- Através da VTL, permite a inserção de informações de maneira dinâmica dentro do *template*
- A VTL possui recursos que permitem:
  - ✓ Referenciar variáveis dentro do *template*
  - ✓ Utilizar controles de fluxo de execução
  - ✓ Definir e invocar macros
  - ✓ Efetuar a chamada de métodos de objetos Java

DB2 Everyplace

- Menor banco de dados do mundo e ocupa aproximadamente 100k de espaço
- Foi desenvolvido para ser utilizado em dispositivos de baixo custo, com pouco poder de processamento e com poucos recursos gráficos
- Os dados podem ser sincronizados com outros bancos de dados DB2 e até mesmo com banco de dados de outros fabricantes
- Possui várias funcionalidades interessantes como suporte a:
  - ✓ Visões
  - ✓ Gatilhos (*triggers*)
  - ✓ Sub-consultas (*triggers*)
  - ✓ Procedimentos armazenados (*Stored Procedures*)
  - ✓ Funções

Desenvolvimento do trabalho

- Especificação dos requisitos funcionais e não funcionais
- Estudo da plataforma JME
- Escolha da forma de armazenamento dos dados nos PDA's
- Interpretação das informações de entrada
- Especificação da saída
- Definição do Tomcat como servidor para importação/exportação
- Especificação da ferramenta através da UML
- Implementação
- Elaboração dos *templates*

Requisitos funcionais

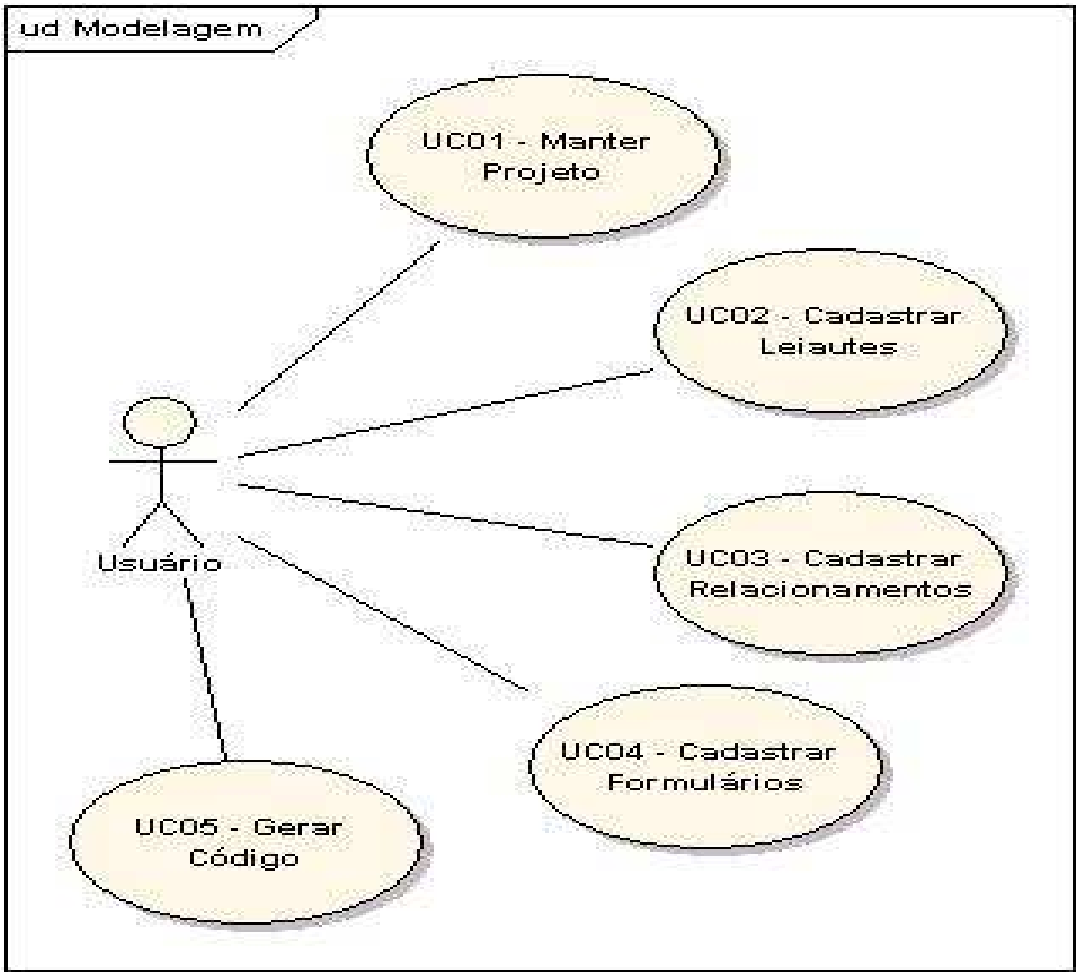
- Permitir cadastrar os leiautes em uma aplicação *desktop*
- Permitir que sejam cadastradas as interfaces de coletas de dados
- Permitir que sejam geradas aplicações JME, contendo itens de menu para cada um dos leiautes associados a interfaces de coletas de dados
- Disponibilizar na aplicação JME a criação das tabelas relacionadas à interface de coleta, logo na primeira vez em que o software seja executado no dispositivo móvel
- Disponibilizar um recurso de importação de dados para os PDAs
- Disponibilizar para que sejam feitas coletas de dados nos dispositivos móveis através do programa gerado e armazenados num banco de dados
- Permitir que os dados coletados nos dispositivos móveis sejam transferidos entre dispositivo móvel e computador

Requisitos não funcionais

- Utilizar a linguagem Java
- Gerar programas para dispositivos móveis na linguagem Java para a plataforma JME
- Programas gerados devem utilizar o banco de dados DB2 Everyplace;
- Disponibilizar nos dispositivos móveis interfaces de boa usabilidade.

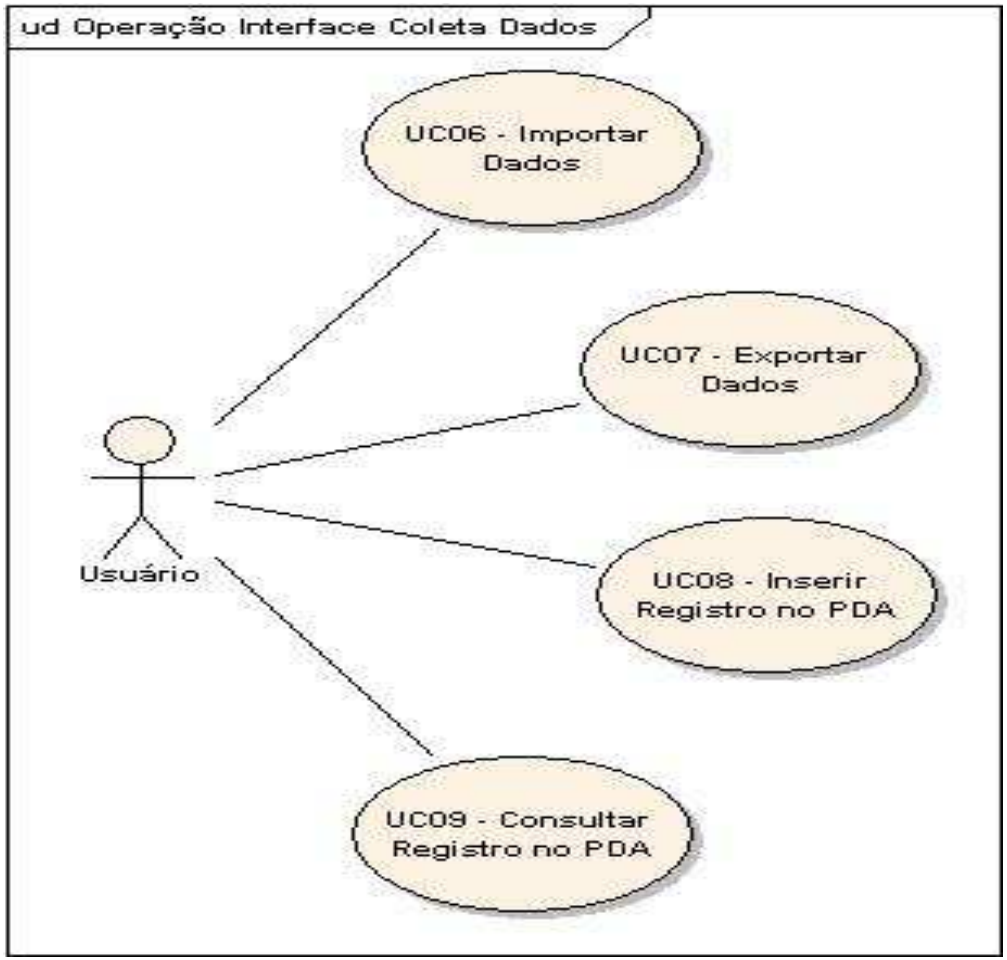
Especificação

- Casos de uso da ferramenta:



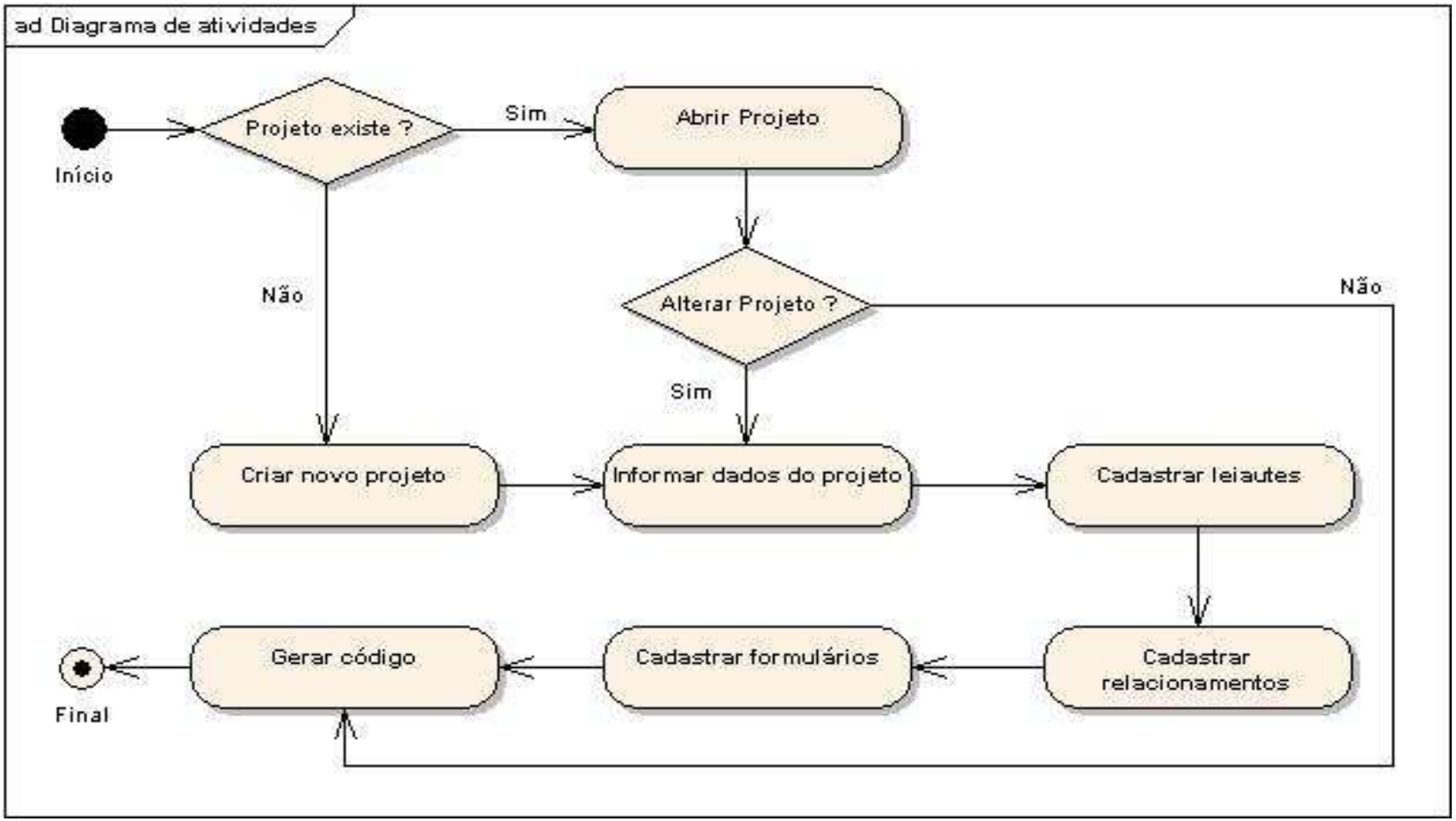
Especificação

- Casos de uso dos programas gerados:

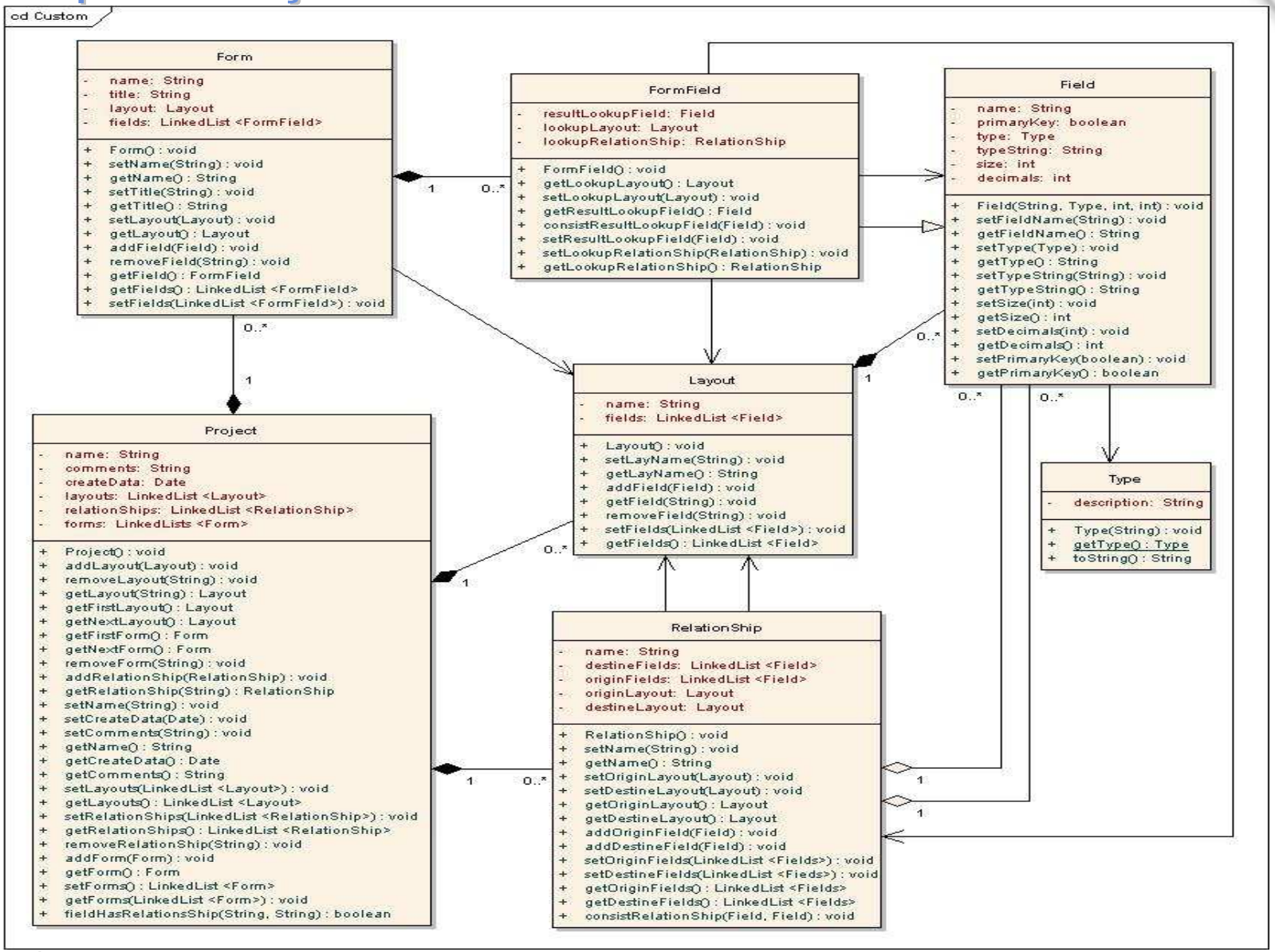


Especificação

- Diagrama de atividades das operações básicas da ferramenta:



Especificação



Você está em: Desenvolvimento do trabalho → Especificação

Implementação

- **Técnicas e ferramentas utilizadas**:
  - **Ferramenta**:
    - ✓ Arquitetura MVC
    - ✓ Java 5.0
    - ✓ NetBeans 5.5
    - ✓ API JDBC
    - ✓ SWING
    - ✓ Velocity
    - ✓ *Templates*
  - **Programas gerados**:
    - ✓ Plataforma J2ME
    - ✓ J2ME Wireless Toolkit
    - ✓ Banco de dados DB2 Everyplace

Implementação

- Método que demonstra a utilização do padrão MVC:

```
private void jbAbrirActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        JFileChooser chooser = new JFileChooser();
        MyFileFilter fileFilter = new MyFileFilter();
        fileFilter.addExtension("xml");
        fileFilter.setDescription("Arquivos XML");
        chooser.setFileFilter(fileFilter);
        chooser.showOpenDialog(this);
        String path = chooser.getSelectedFile().getPath();
        if  (!(path.equals(" "))) {
            HashMap <String, String> values = iController.getProject(path);
            this.setName(values.get(iController.PROJECT_NAME));
            SimpleDateFormat simpleFormat = new SimpleDateFormat("dd/MM/yyyy");
            this.setCreateData(simpleFormat.parse(values.get(iController.PROJECT_CREATEDATA)));
            this.setComments(values.get(iController.PROJECT_COMMENTS));
            this.setAddingItems(true);
            this.chargeJcbLayoutName();
            this.setAddingItems(false);
            this.setLayName(values.get(iController.PROJECT_FIRST_LAYOUT));
            this.chargeLayout(this.getLayName());
            this.enableDefaultControls(true);
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, "Não foi possível abrir o projeto: "+e.getMessage(),
                "Erro", JOptionPane.ERROR_MESSAGE);
    }
```

Implementação

- Método que efetua a geração dos códigos dos formulários:

```
public static void generateForms(Project project)
throws ResourceNotFoundException, ParseErrorException, MethodInvocationException,
        IOException, Exception {

    String path = "./src/templates";
    String nameTemplate = "TemplateForm.vm";

    Convert convert = new Convert();
    Map<String, Object> context = new HashMap<String, Object>();
    String fullPath;

    for (Form form : project.getForms()) {
        fullPath = "../PDAMagRegister/src/Frm" + convert.convertUpperFirst(form.getName()) + ".java";
        context.put( "project", project );
        context.put( "convert", convert );
        context.put( "form", form);
        VelocityController.generate( path, nameTemplate, context, fullPath );
    }

    context.clear();
    nameTemplate = "TemplateFrmPesqRegistros.vm";
    fullPath = "../PDAMagRegister/src/FrmPesqRegistros.java";
    VelocityController.generate( path, nameTemplate, context, fullPath );

}
```

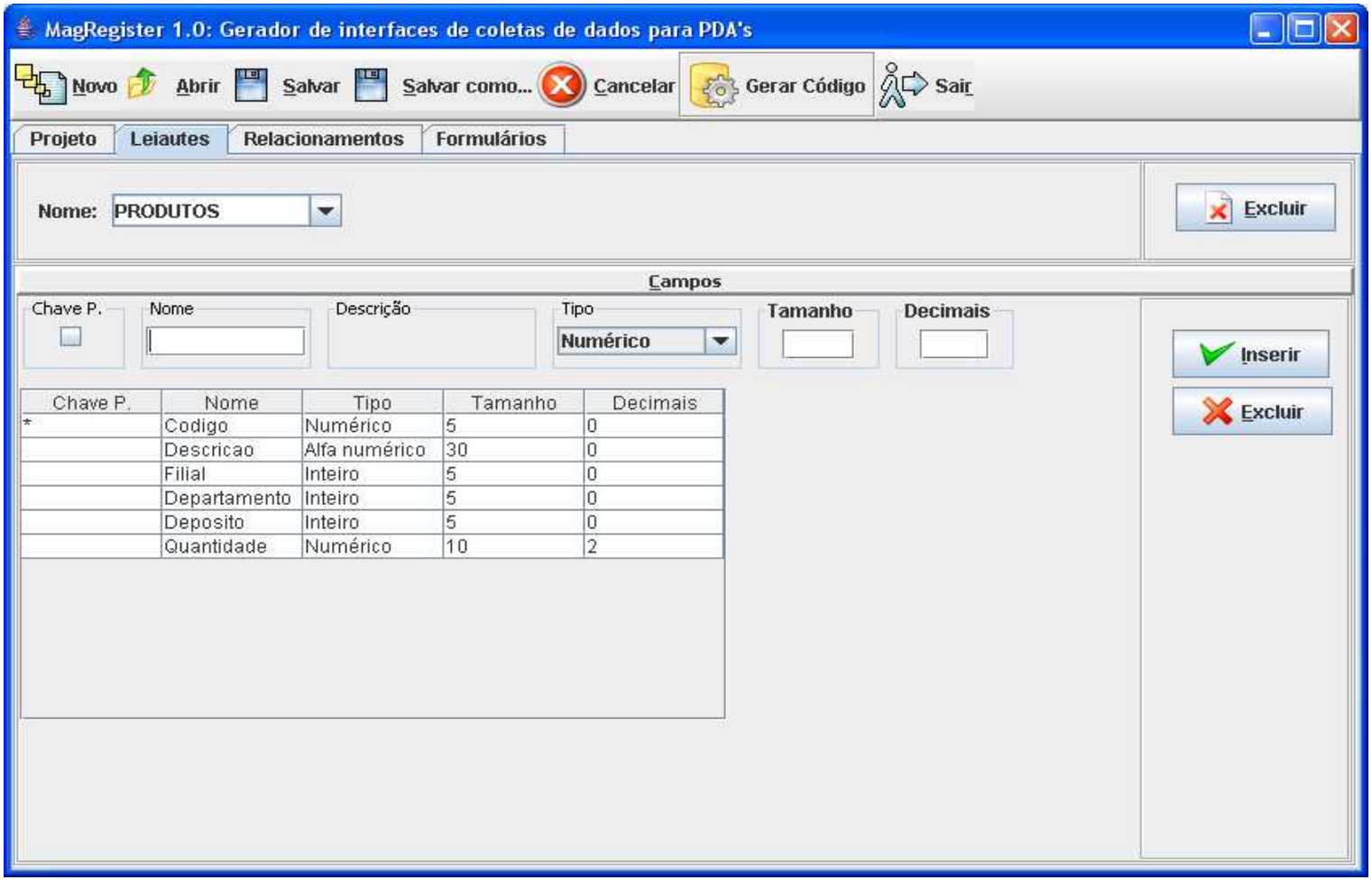
Implementação

- *Template* utilizado na geração do menu principal de aplicação:

```
import javax.microedition.lcdui.Command;
.
public class MainMenu extends Form implements CommandListener{

    #foreach($form in $project.getForms())
    private Command cmd$convert.convertUpperFirst($form.getName());
    #end

    private Command cmdDivision1;
    .
    .
    private MainMidlet midlet;
    private Display display; // Formsr

    public MainMenu(MainMidlet mainMidlet) {
        super("Escolha a opção");
        display = mainMidlet.getDisplay();
        midlet = mainMidlet;
        #foreach($form in $project.getForms())
        cmd$convert.convertUpperFirst($form.getName()) = new Command("$convert.convertUpperFirst($form.getName())", Command.ITEM, 1);
        #end
        cmdDivision1 = new Command("---------------", Command.ITEM, 1);
        cmdDivision2 = new Command("---------------", Command.ITEM, 1);
        cmdDropTables = new Command("Excluir Tabelas", Command.ITEM, 1);
        cmdImportar = new Command("Importar Dados", Command.ITEM, 1);
        cmdExportar = new Command("Exportar Dados", Command.ITEM, 1);
        cmdSair = new Command("Sair", Command.ITEM,1);
        #foreach($form in $project.getForms())
        addCommand(cmd$convert.convertUpperFirst($form.getName()));
        #end
        addCommand(cmdDivision1);
        .
        .
        setCommandListener(this);
    }
```

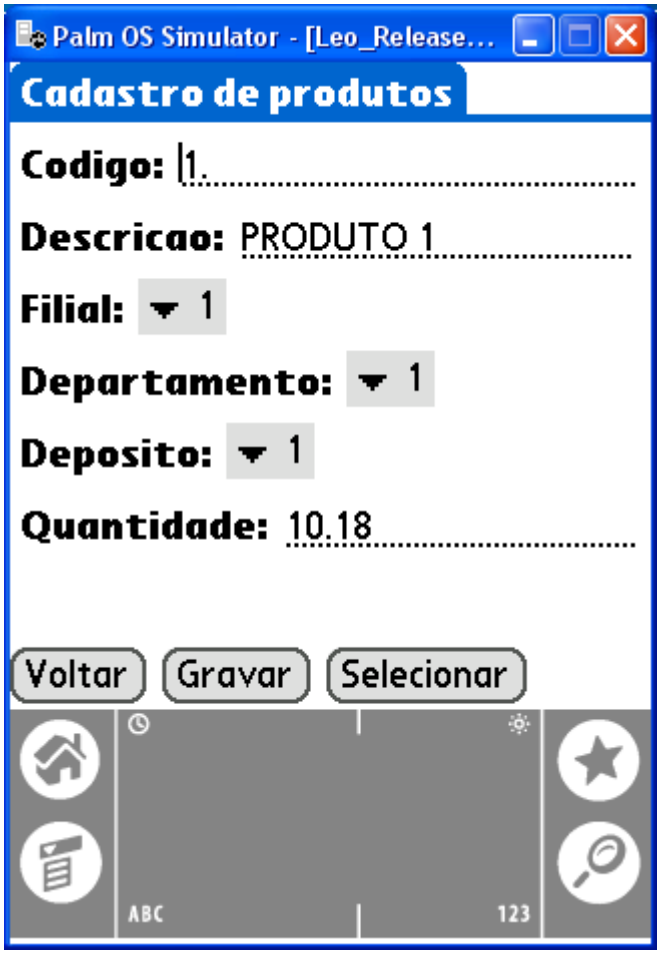
Operacionalidade

- *Tela de cadastramento dos leiautes*:



Operacionalidade

- *Interface de coleta de dados para PDA*:



Resultados e discussão

- Atendimento dos requisitos propostos
- Motor de *templates* Velocity facilitou bastante a geração de códigos
- Testes efetuados no simulador do Palm OS Tungsten E2
- Comparativo entre as ferramentas:

CARACTERÍSTICAS	MAG REGISTER	SILVA (2005)	LESSNAU (2002)	DEPINÉ (2002)
Geração de programas de forma customizável	X		X	
Banco de dados no dispositivo móvel	X	X		
Utilização plataforma JME	X	X	X	X
Telas no formato mestre-detalhe		X	X	

Conclusão

- Funcionalidades
  - ✓ Ferramenta que pode abranger várias áreas de negócio
  - ✓ Pode se tornar um produto, caso seja dada continuidade
  - ✓ Auxilia no processo de coleta de dados externos a um software de gestão
- Flexibilidade
  - ✓ Código gerado definido em *templates*
  - ✓ Portabilidade da ferramenta e dos aplicativos gerados
- Aprendizado
  - ✓ Tecnologias e padrões totalmente desconhecidos
- Limitações
  - ✓ Quantidade de campos na chave primária
  - ✓ URL de conexão com o *servlet* para troca de dados e templates pré-fixados na ferramenta

Extensões

- Leiautes mestre-detalhes
- Troca de dados entre PC e PDAs via cabos
- Modelagem de formato de arquivos textos para a troca de dados

Obrigado !!!