Controle de vacinas e imunizações utilizando Smart Cards

Eduardo Paniz Mallmann

Orientado por Marcel Hugo

Roteiro da apresentação

- Introdução
- Objetivos
- Fundamentação Teórica
- Requisitos
- Implementação
- Desenvolvimento
- Resultados e Discussão
- Conclusão
- Extensões

Introdução

- Atual modelo da carteira de saúde
- Solução proposta

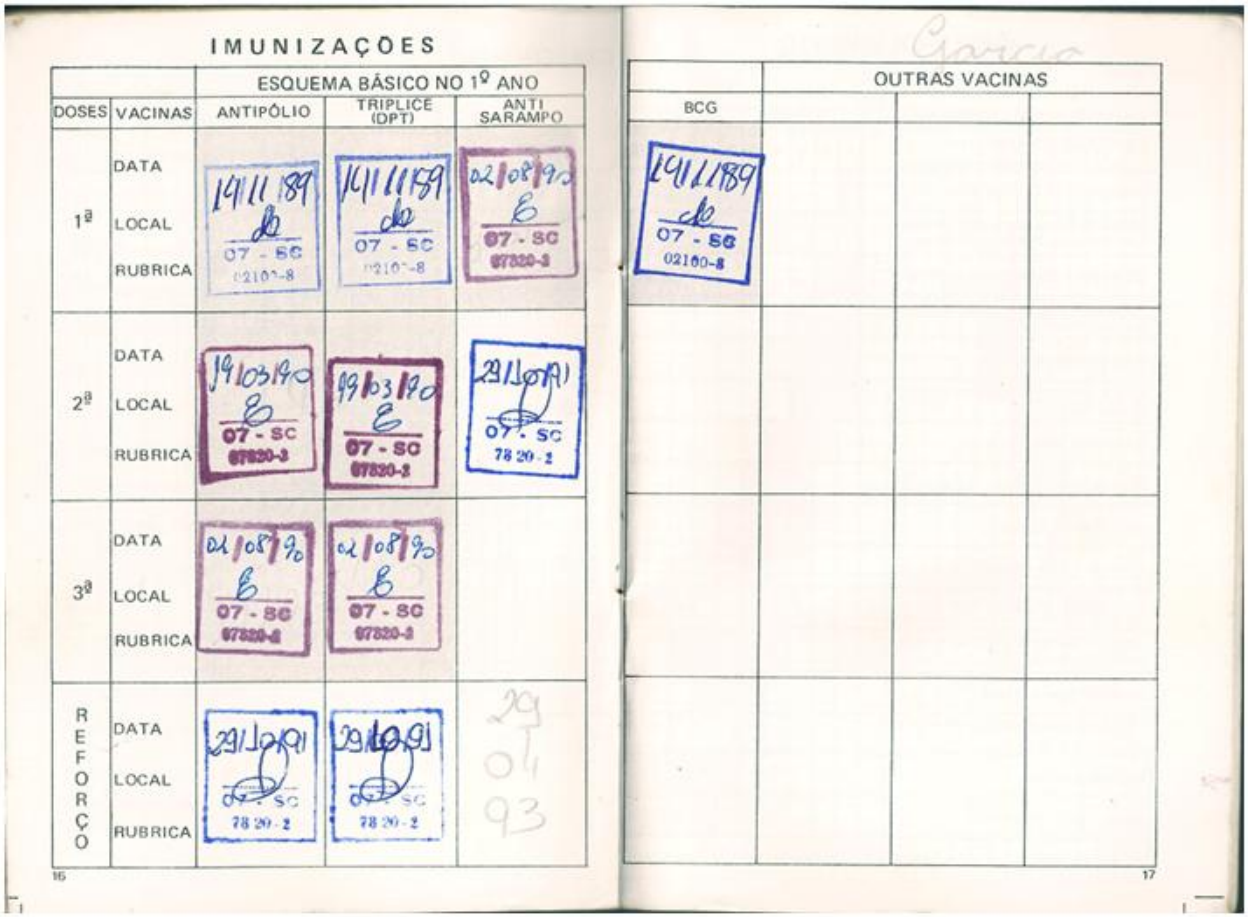
Objetivos

- Utilizar métodos atuais de criptografia para garantir a segurança e integridade das informações
- Desenvolver um sistema para registrar novas imunizações e vacinas no smart card que será utilizado pelos agentes de saúde
- Desenvolver um sistema para visualização das informações contidas no smart card, para ser utilizado pelo cidadão dono do cartão.

Fundamentação Teórica

- Caderneta de Saúde
- *Smart cards*
- Java *Card*
- APDU
- Trabalhos correlatos

Caderneta de Saúde



IMUNIZAÇÕES

DOSES	VACINAS	ESQUEMA BÁSICO NO 1º ANO – ANTIPÓLIO	ESQUEMA BÁSICO NO 1º ANO – TRIPLICE (DPT)	ESQUEMA BÁSICO NO 1º ANO – ANTI SARAMPO
1ª	DATA	14/11/89	14/11/89	02/08/90
	LOCAL	07 - SC 02160-8	07 - SC 02160-8	07 - SC 07320-2
	RUBRICA			
2ª	DATA	19/03/90	19/03/90	29/10/91
	LOCAL	07 - SC 07320-2	07 - SC 07320-2	07 - SC 78 20 - 2
	RUBRICA			
3ª	DATA	02/08/90	02/08/90	
	LOCAL	07 - SC 07320-2	07 - SC 07320-2	
	RUBRICA			
REFORÇO	DATA	29/10/91	29/10/91	29/04/93
	LOCAL	07 - SC 78 20 - 2	07 - SC 78 20 - 2	
	RUBRICA			

16

BCG	OUTRAS VACINAS		
14/11/89 07 - SC 02160-8			

17

Caderneta de Saúde

- Documento de fase infantil até fase idosa
- Caderneta de Saúde da Criança, do Adolescente e da Pessoa Idosa
- Distribuída em postos de saúde do Brasil e maternidades de hospitais

Smart Cards


JCOP
ID 21
sonsun
www.sonsun.com.br
JCOP
ID 21
Java
ENGINEERING SAMPLE
JAVACARD 2.2.1 - 36K

Smart Cards

- Dispositivos móveis com chip para processamento e armazenamento
- Três tipos de memória: ROM, RAM e EEPROM
- Acessados a partir de um CAD (Leitora)
- ISO 7816

Java Card

- Tecnologia Java para ser utilizada em dispositivos móveis que possuem memória e processamento limitados
- Especificação mais atual: 3.0
- Classic Edition e Connected Edition
- Java Card Virtual Machine
- Java Card Runtime Environmnent

APDU

- Application Protocolo Data Unit
- Trocar mensagens entre aplicação local e smart card
- Estrutura de envio e de resposta

APDU

Command APDU

Cabeçalho obrigatório				Corpo opcional		
CLA	INS	P1	P2	Lc	Datafield	Le

Response APDU

Corpo opcional	Trailer	
Datafield	SW1	SW2

Trabalhos Correlatos

- Documentos e Dinheiro Eletrônico com Smart Card Utilizando Tecnologia Java Card. Cleber Giovanni Suavi

- Smart Interface: Ferramenta de Auxílio ao Desenvolvimento de Aplicações Java Card. Minora, Aleixo, Diolino

Requisitos

- Desenvolver uma interface para seleção do leitor de cartão
- Desenvolver uma interface para cadastrar ou alterar os dados pessoais de um portador do cartão
- Desenvolver uma interface para cadastro de imunizações e vacinas para serem utilizadas no sistema
- Desenvolver uma interface para registar imunizações e vacinas de um determinado portador

Requisitos

- Desenvolver um sistema visualizador das informações contidas no smart card
- Implementar rotina de criptografia nas informações para garantir sua integridade e segurança
- Ser implementado na linguagem de programação Java SE 6
- Utilizar banco de dados MySql

Especificação

- Casos de uso, diagrama de atividades e diagrama de classes
- Enterprise Architect (UML)
- DBDesigner (MER)

Casos de uso

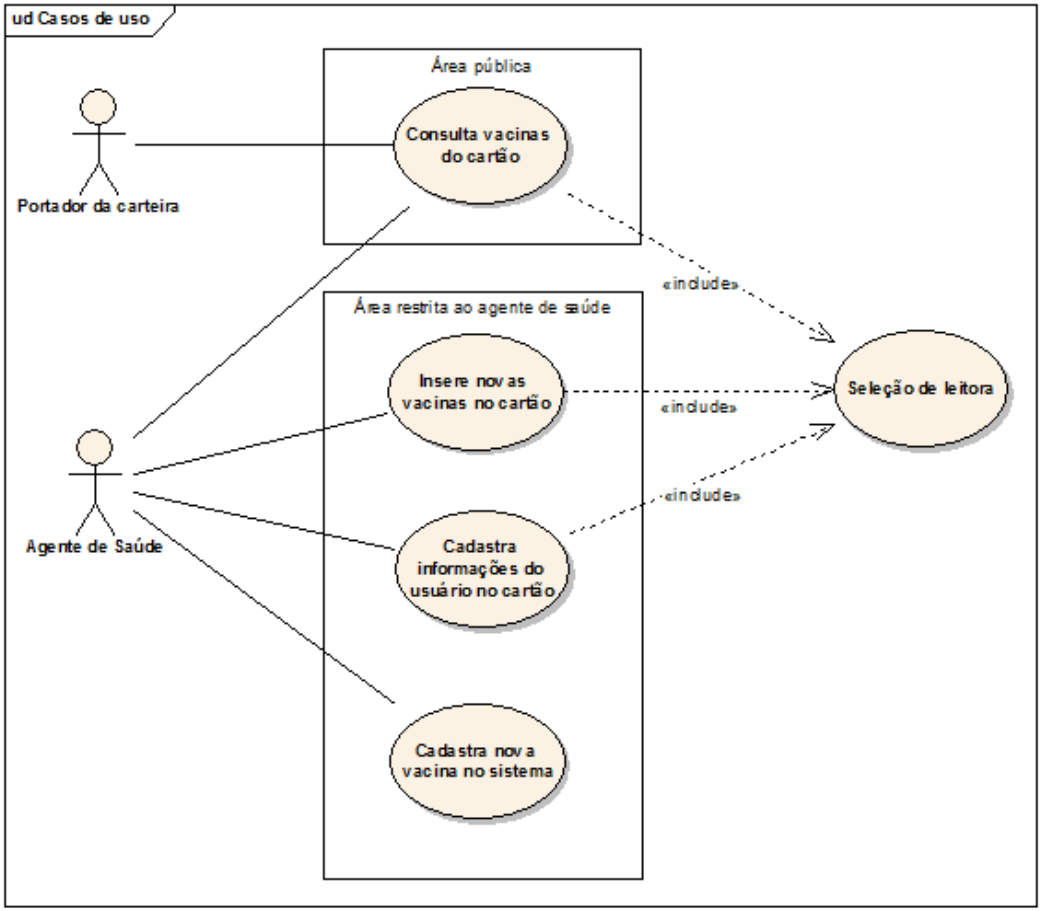

ud Casos de uso
Área pública
Consulta vacinas do cartão
Portador da carteira
«include»
Área restrita ao agente de saúde
Insere novas vacinas no cartão
Seleção de leitora
«include»
Agente de Saúde
«include»
Cadastra informações do usuário no cartão
Cadastra nova vacina no sistema

Diagrama de atividades

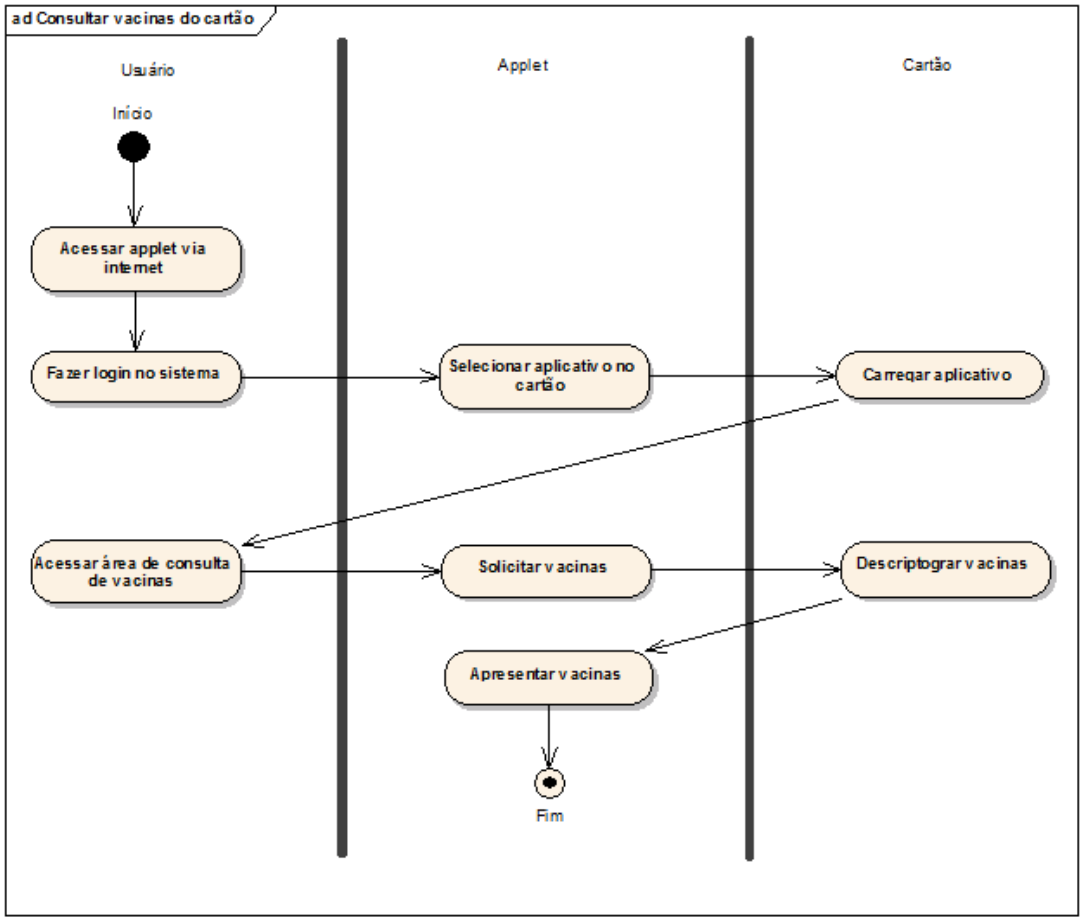


Diagrama de atividades

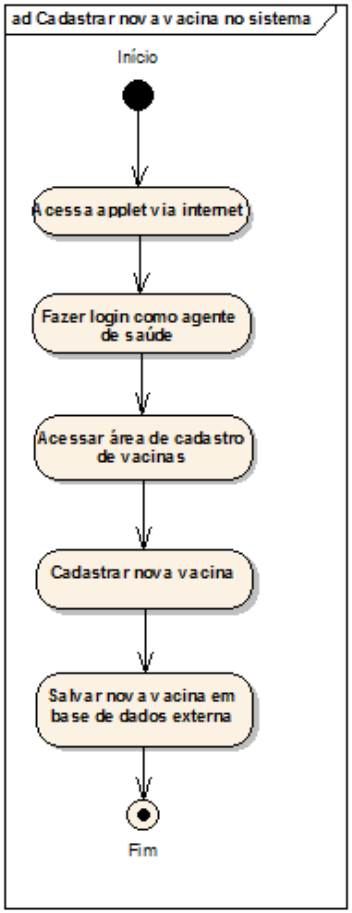


Diagrama de atividades

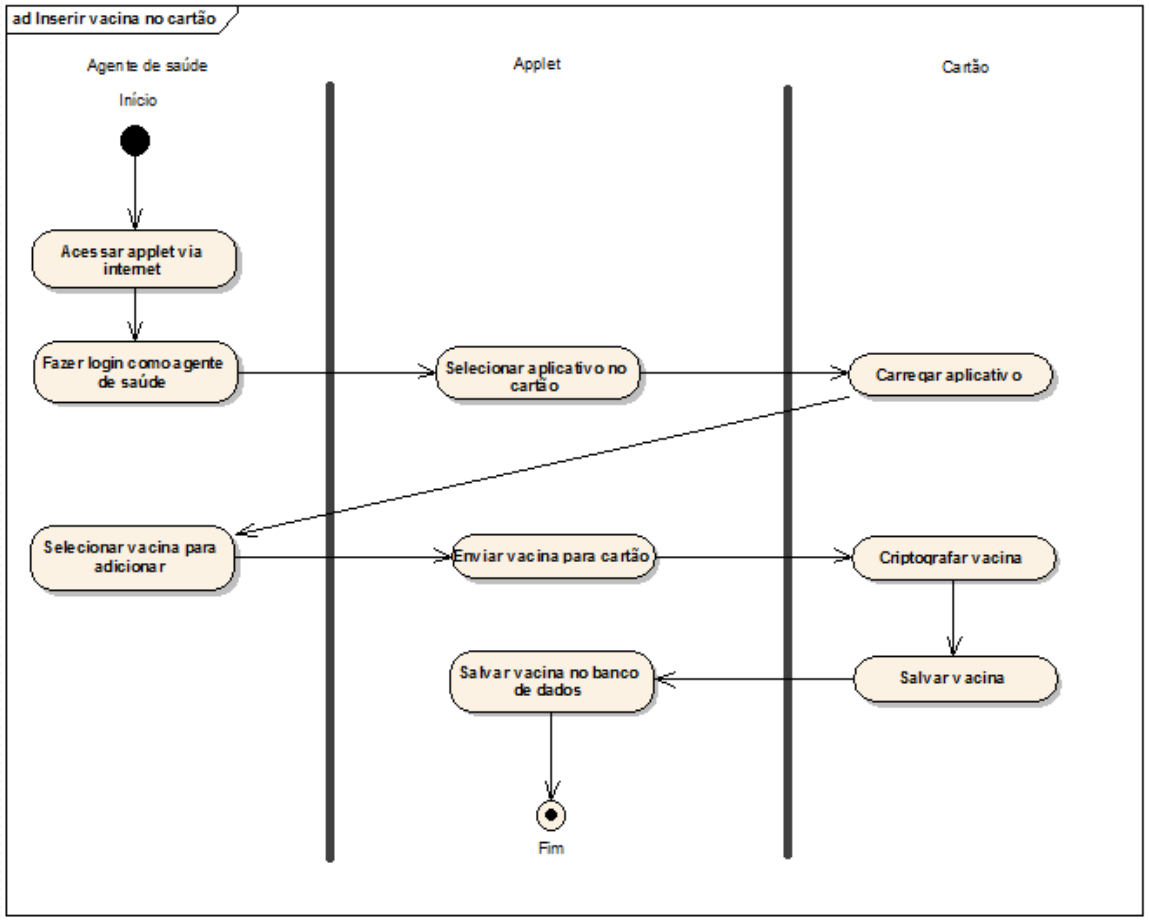


Diagrama de atividades

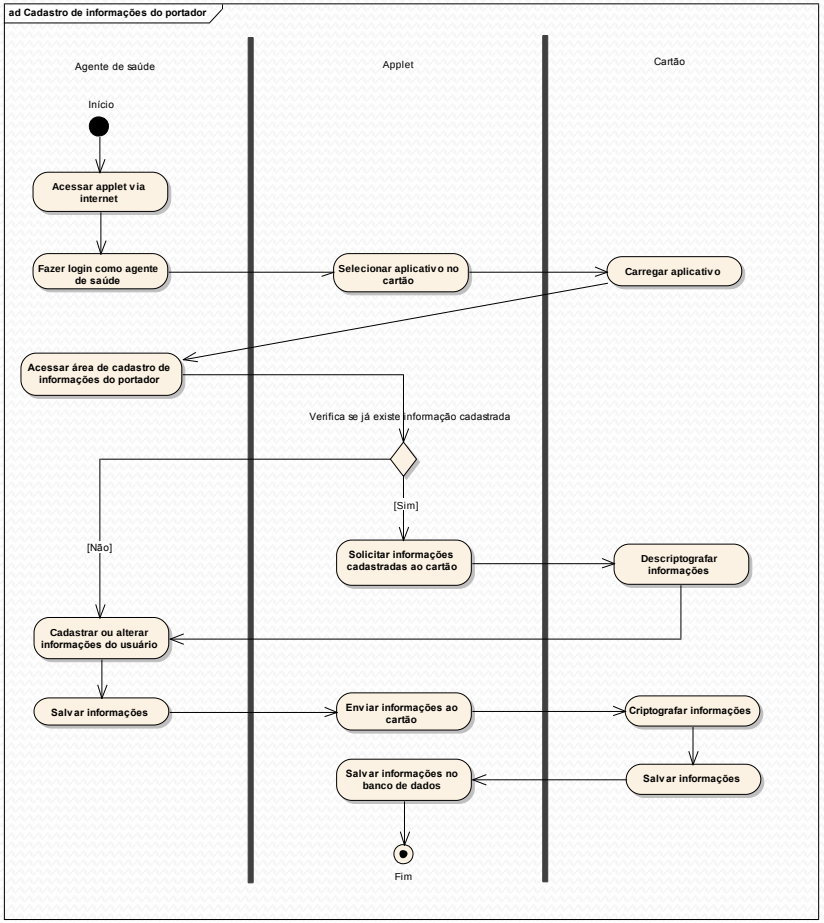


Diagrama de classes

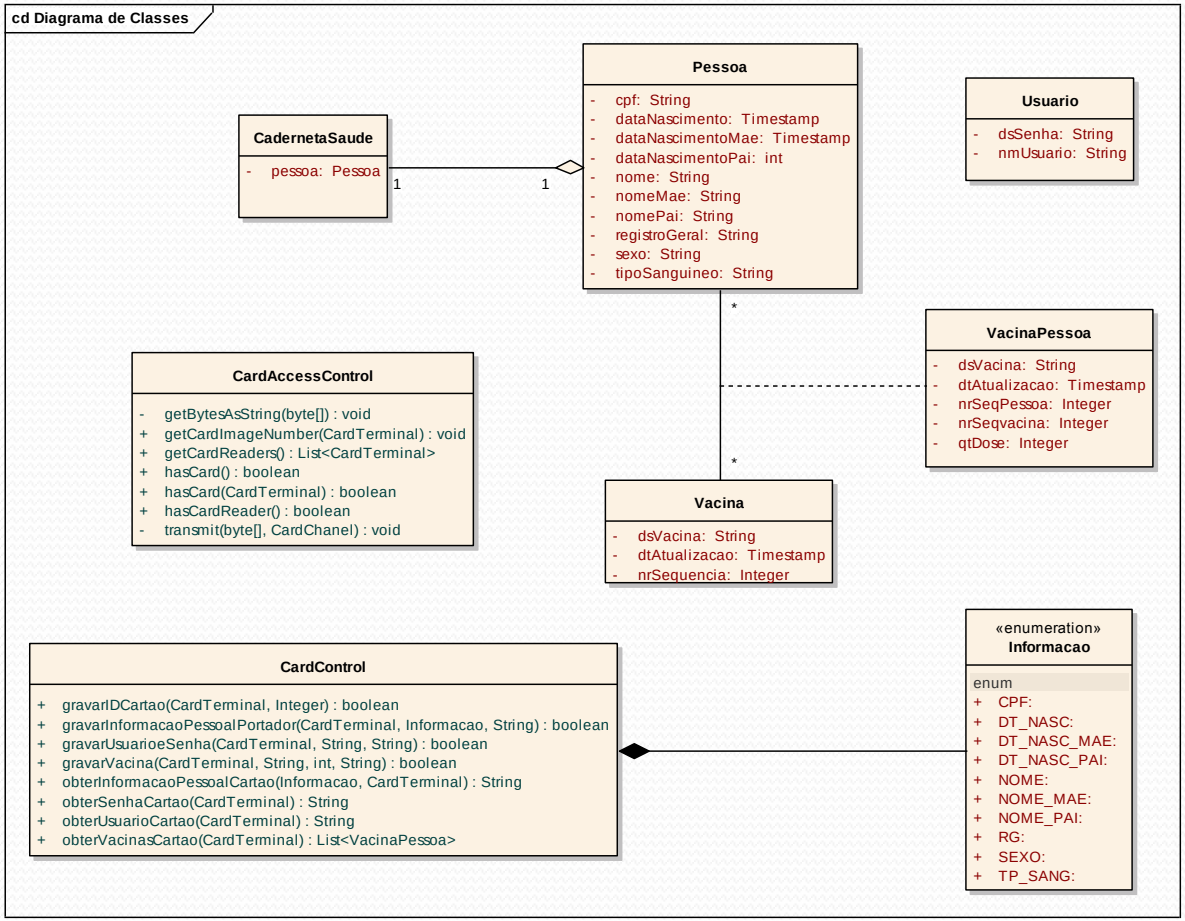


Diagrama de classes

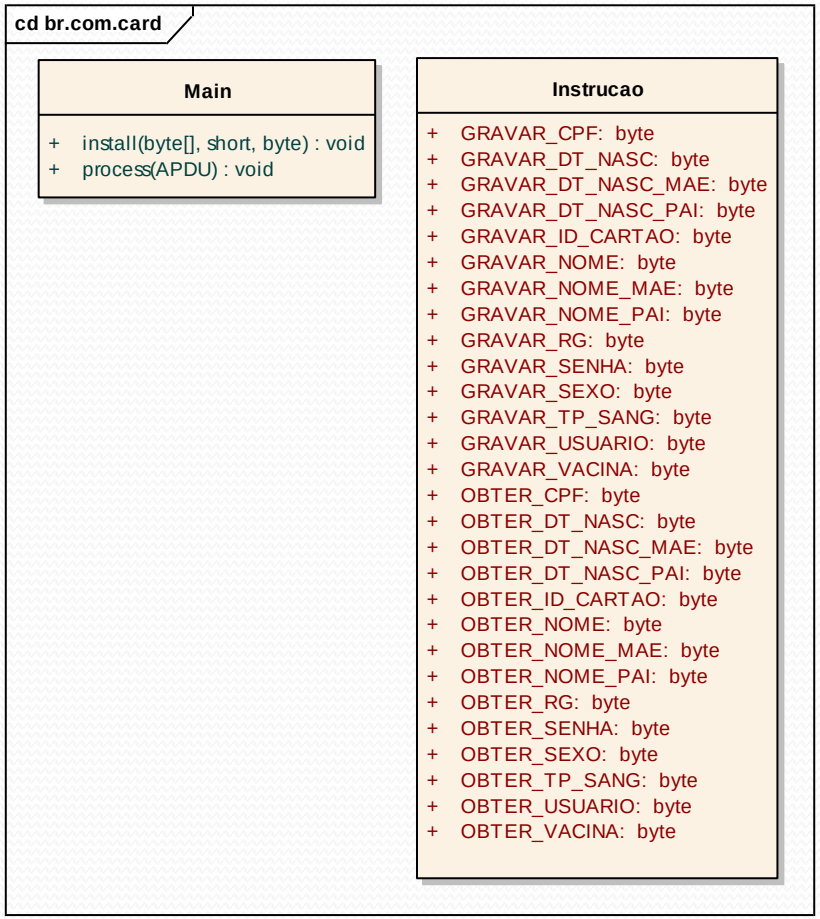


Diagrama de classes

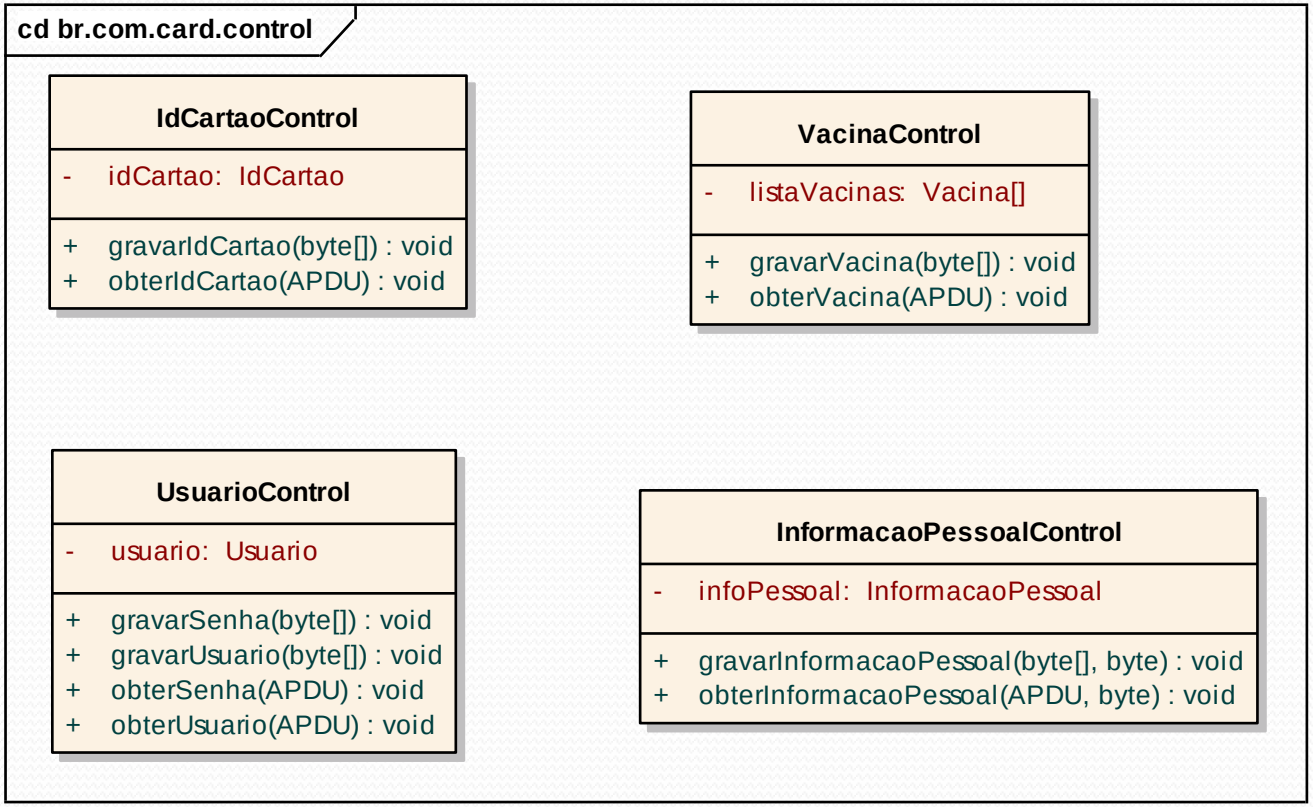


Diagrama de classes

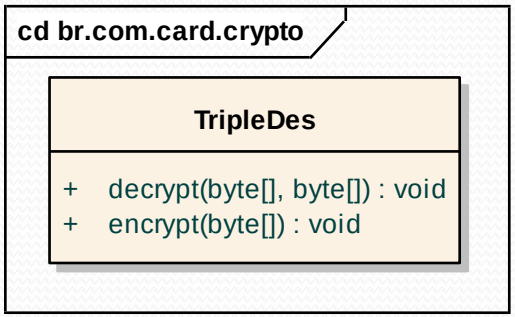
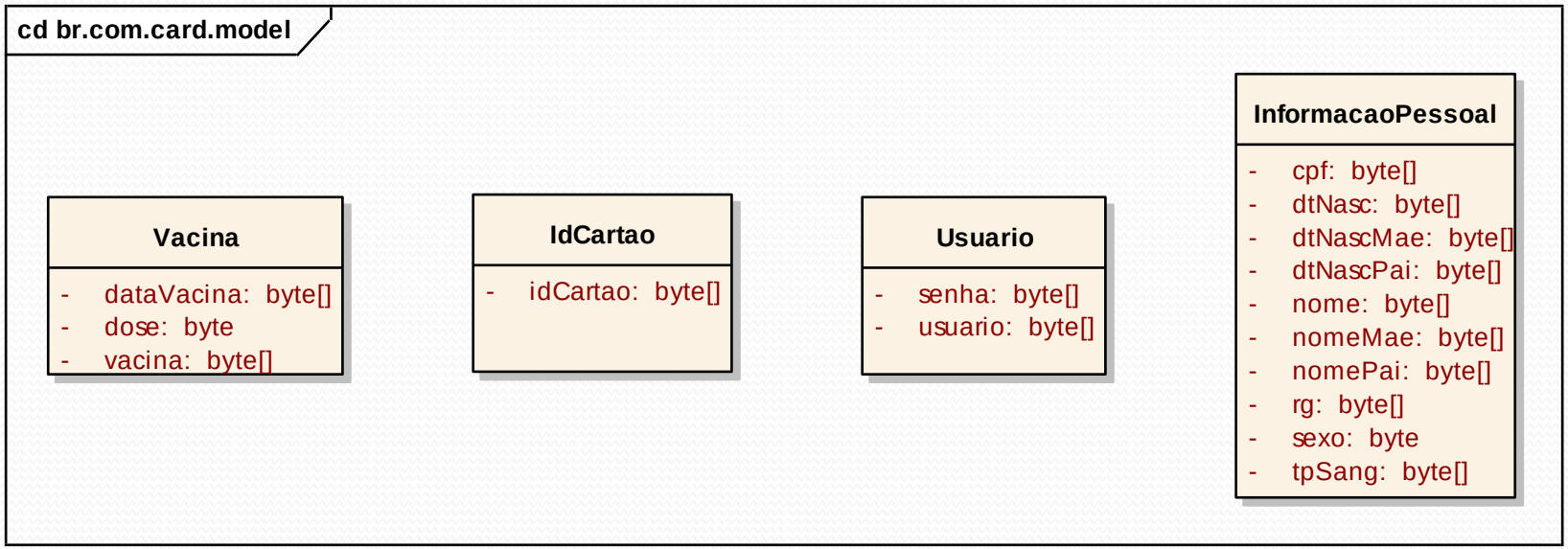
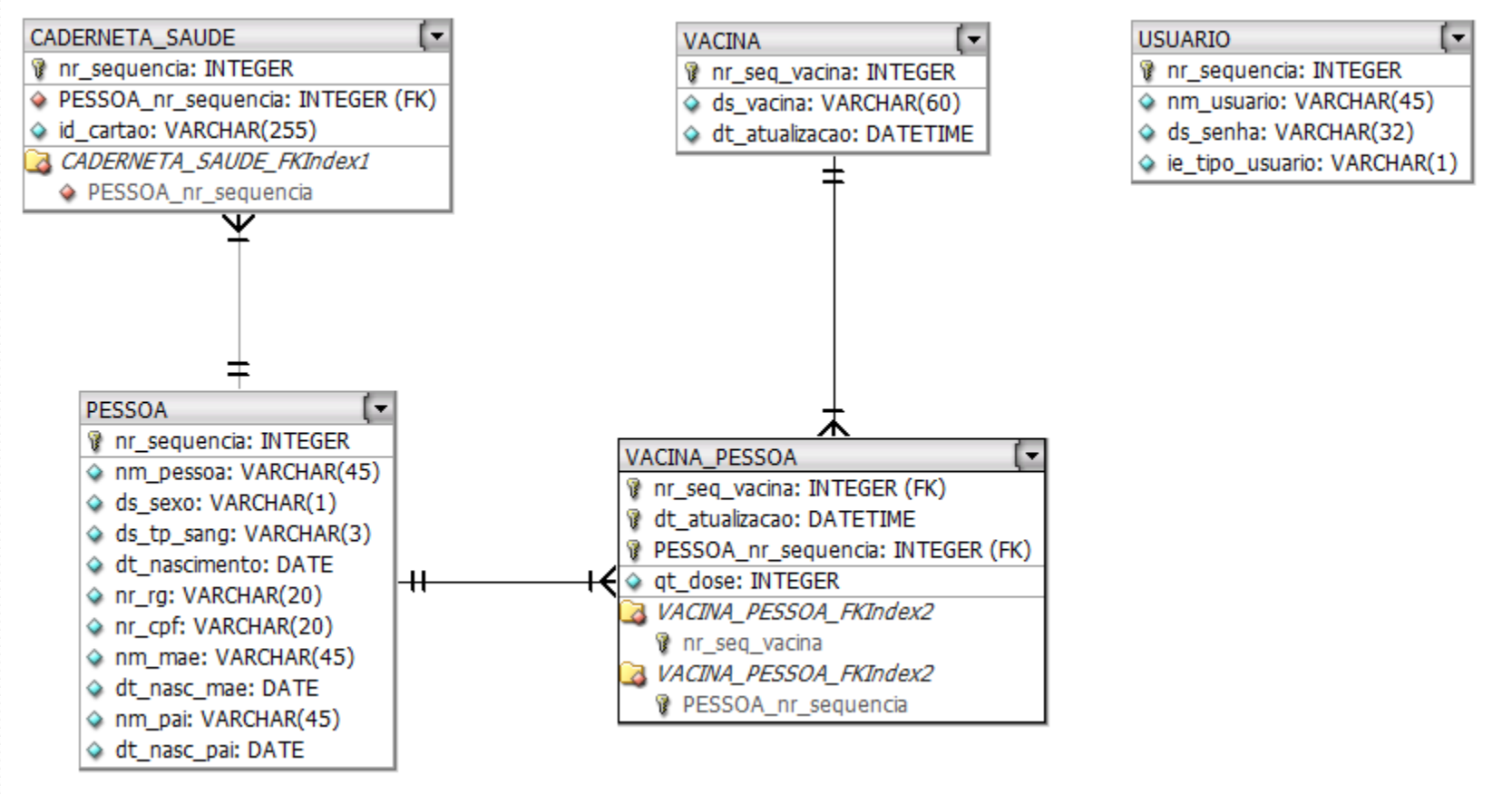


Diagrama de classes



MER


CADERNETA_SAUDE
nr_sequencia: INTEGER
PESSOA_nr_sequencia: INTEGER (FK)
id_cartao: VARCHAR(255)
CADERNETA_SAUDE_FKIndex1
PESSOA_nr_sequencia
VACINA
nr_seq_vacina: INTEGER
ds_vacina: VARCHAR(60)
dt_atualizacao: DATETIME
USUARIO
nr_sequencia: INTEGER
nm_usuario: VARCHAR(45)
ds_senha: VARCHAR(32)
ie_tipo_usuario: VARCHAR(1)
PESSOA
nr_sequencia: INTEGER
nm_pessoa: VARCHAR(45)
ds_sexo: VARCHAR(1)
ds_tp_sang: VARCHAR(3)
dt_nascimento: DATE
nr_rg: VARCHAR(20)
nr_cpf: VARCHAR(20)
nm_mae: VARCHAR(45)
dt_nasc_mae: DATE
nm_pai: VARCHAR(45)
dt_nasc_pai: DATE
VACINA_PESSOA
nr_seq_vacina: INTEGER (FK)
dt_atualizacao: DATETIME
PESSOA_nr_sequencia: INTEGER (FK)
qt_dose: INTEGER
VACINA_PESSOA_FKIndex2
nr_seq_vacina
VACINA_PESSOA_FKIndex2
PESSOA_nr_sequencia

Implementação – Applet Web

- Java SE 6
- Tomcat 6.0.20
- HTML (Visualização applet)
- NetBeans 6.8

Implementação – Smart Card

- Java Card 2.2.1
- Eclipse Galileo

Desenvolvimento web applet

- javax.smartcardio

- TerminalFactory
- CardTerminal
- Card
- CardException
- CommandAPDU
- ResponseAPDU

Exemplo

```
public static boolean hasCard() {
    try {
        CardTerminals cts = TerminalFactory.getDefault().terminals();
        if (cts != null && cts.list() != null && cts.list().size() > 0) {
            CardTerminal terminal = cts.list().get(0);
            return terminal.isCardPresent();
        }
        return false;
    } catch (CardException ex) {
        if (Config.isDebug()) {
            Logger.getLogger(CardAccessControl.class.getName()).log(Level.SEVERE, null, ex);
        }
        return false;
    }
}
```

Comando APDU

Função	CLA	INS	P1	P2	LC	Dados	LE
Gravar vacina	0x00	0x10	0x00	0x00	Tamanho dados	Vacina 0x00 dose 0x00 data	-

```java
public boolean gravarVacina(CardTerminal cardTerminal, String vacina, int dose, String data) {
    try {
        if (dose == -1) {
            dose = 0;
        }
        int tamanhoDados = vacina.getBytes().length + 13;
        int tamanhoArray = 5 + tamanhoDados;
        byte[] comandoGravarVacina = new byte[tamanhoArray];
        comandoGravarVacina[0] = 0x00;
        comandoGravarVacina[1] = 0x10;
        comandoGravarVacina[2] = 0x00;
        comandoGravarVacina[3] = 0x00;
        comandoGravarVacina[4] = (byte) tamanhoDados;
        int pos = 5;
        for (int i = 0; i < vacina.getBytes().length; i++) {
            comandoGravarVacina[pos++] = vacina.getBytes()[i];
        }
        comandoGravarVacina[pos++] = 0x00;
        comandoGravarVacina[pos++] = (byte) dose;
        comandoGravarVacina[pos++] = 0x00;
        for (int i = 0; i < data.getBytes().length; i++) {
            comandoGravarVacina[pos++] = data.getBytes()[i];
        }
        Card card = cardTerminal.connect("*");
        CardChannel cc = card.getBasicChannel();
        transmitirComando(cc, comandoGravarVacina);
        card.disconnect(true);
        return true;
    } catch (Exception e) {
        e.printStackTrace();
        return false;
    }
}
```

Response APDU

Função	CLA	INS	P1	P2	LC	Dados	LE
Ob ter vacina	0x00	0x03	0x00 ou 0x01	0x00	-	-	59 bytes

Função	Dados retornados
Obter vacina	Vacina 0x00 dose 0x00 data 0x00 ou 0x01

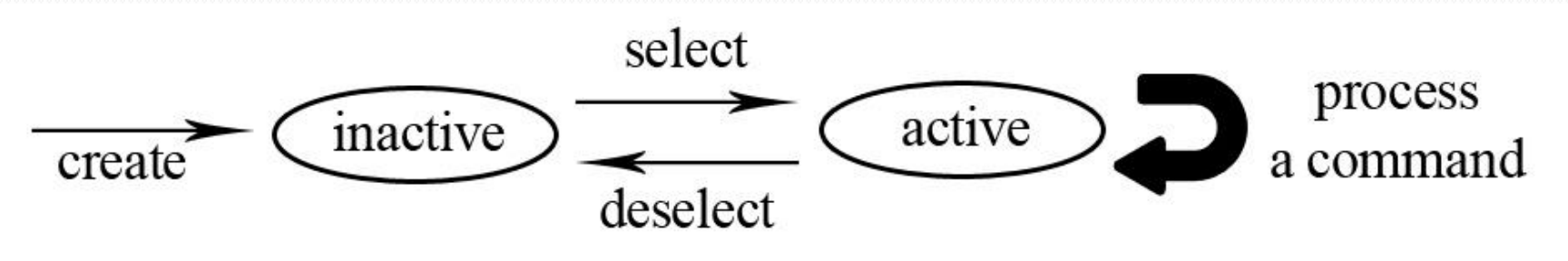
```java
public List<VacinaPessoa> obterVacinasCartao(CardTerminal cardTerminal) {
    List<VacinaPessoa> listaVacinas = new ArrayList<VacinaPessoa>();
    try {
        byte[] comandoObterVacina = new byte[]{0x00, 0x03, 0x00, 0x00, 0x3A};

        Card card = cardTerminal.connect("*");
        CardChannel cc = card.getBasicChannel();

        byte temMaisVacina = 0x01;
        while (temMaisVacina == 0x01) {
            ResponseAPDU apdu = transmitirComando(cc, comandoObterVacina);
            byte[] resposta = apdu.getBytes();

            String vacina = String.valueOf(Arrays.copyOfRange(resposta, 0, 44));
            int dose = Integer.parseInt(String.valueOf(Arrays.copyOfRange(resposta, 45, 46)));
            String data = String.valueOf(Arrays.copyOfRange(resposta, 48, 57));
            temMaisVacina = resposta[58];

            VacinaPessoa vacinaPessoa = new VacinaPessoa();
            vacinaPessoa.setDsVacina(vacina);
            vacinaPessoa.setQtDose(dose);
            vacinaPessoa.setDtAtualizacao(Util.getStringAsTimestamp(data));

            listaVacinas.add(vacinaPessoa);

            if (comandoObterVacina[2] == 0x00) {
                comandoObterVacina[2] = 0x01;
            }
        }

        card.disconnect(true);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return listaVacinas;
}
```

Desenvolvimento Smart Card

- javacard.framework
- javacard.security

- Applet
- APDU
- ISO7816
- Util
- DESKey
- Cipher

Estados



Principais métodos

- install
- select
- process
- deselect

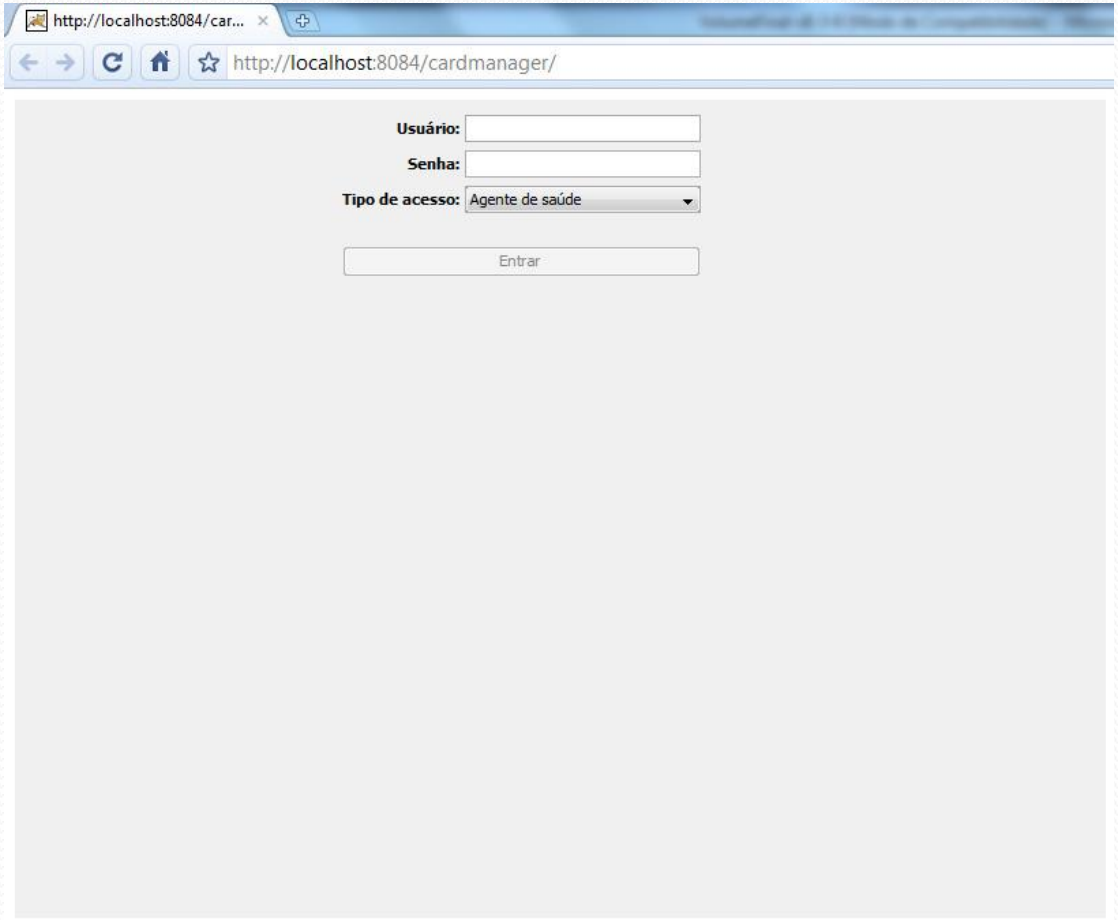
Exemplo estrutura

```
1  package br.com.card;
2
3  import javacard.framework.APDU;
4  import javacard.framework.Applet;
5  import javacard.framework.ISOException;
6
7  public class ExemploEstruturaApplet extends Applet {
8
9      protected ExemploEstruturaApplet() {
10         register();
11     }
12
13     public static void install(byte[] bArray, short bOffset, byte bLength) {
14         new ExemploEstruturaApplet();
15     }
16
17     public boolean select() {
18         return true;
19     }
20
21     public void process(APDU apdu) throws ISOException {
22     }
23
24     public void deselect() {
25     }
26 }
27
```

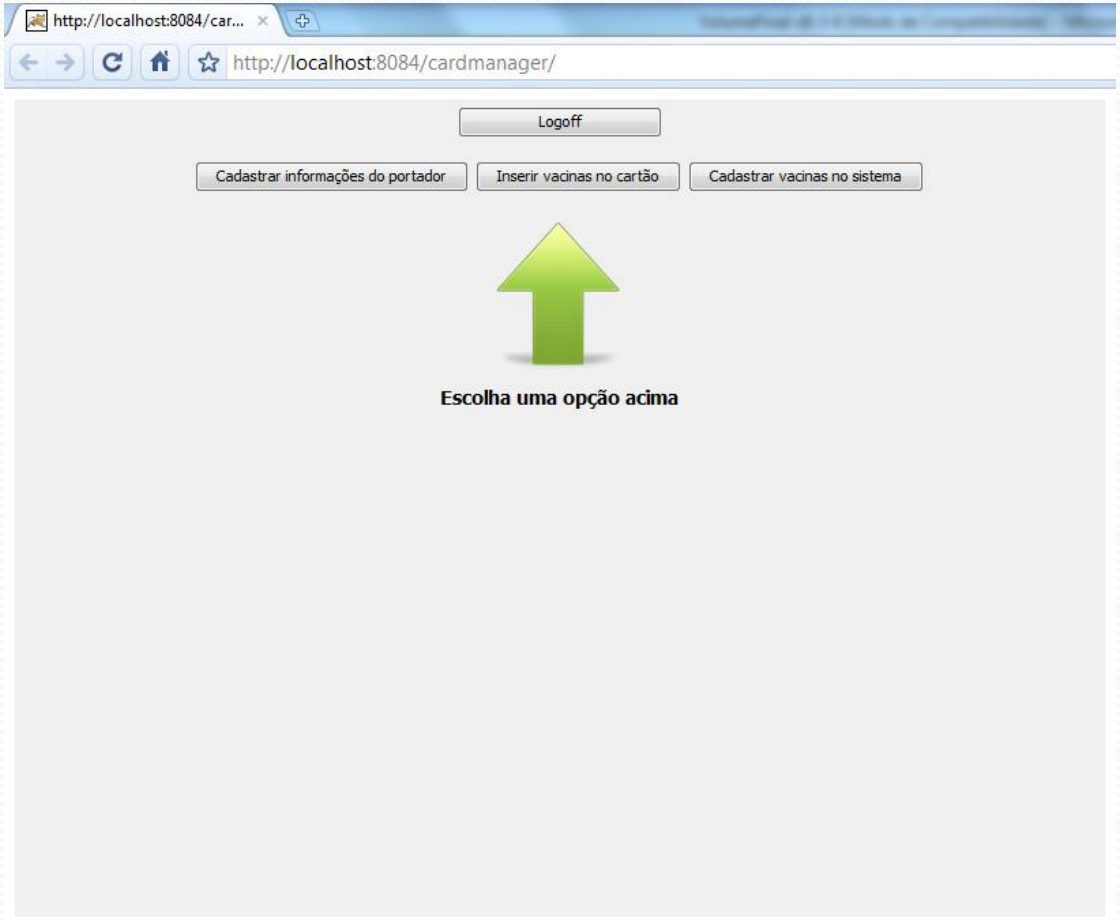
Exemplo leitura e gravação

```
private Usuario usuario = new Usuario();

public void gravarUsuario(byte[] buffer) {
    byte[] user = new byte[16];
    Util.arrayCopy(buffer, (short) 5, user, (short) 0, (short) 16);
    TripleDes.encrypt(user);
    usuario.setUsuario(user);
}

public void obterUsuario(APDU apdu) {
    apdu.setOutgoing();
    apdu.setOutgoingLength((short) 16);
    byte[] user = new byte[16];
    TripleDes.decrypt(usuario.getUsuario(), user);
    apdu.sendBytesLong(user, (short) 0, (short) user.length);
}
```

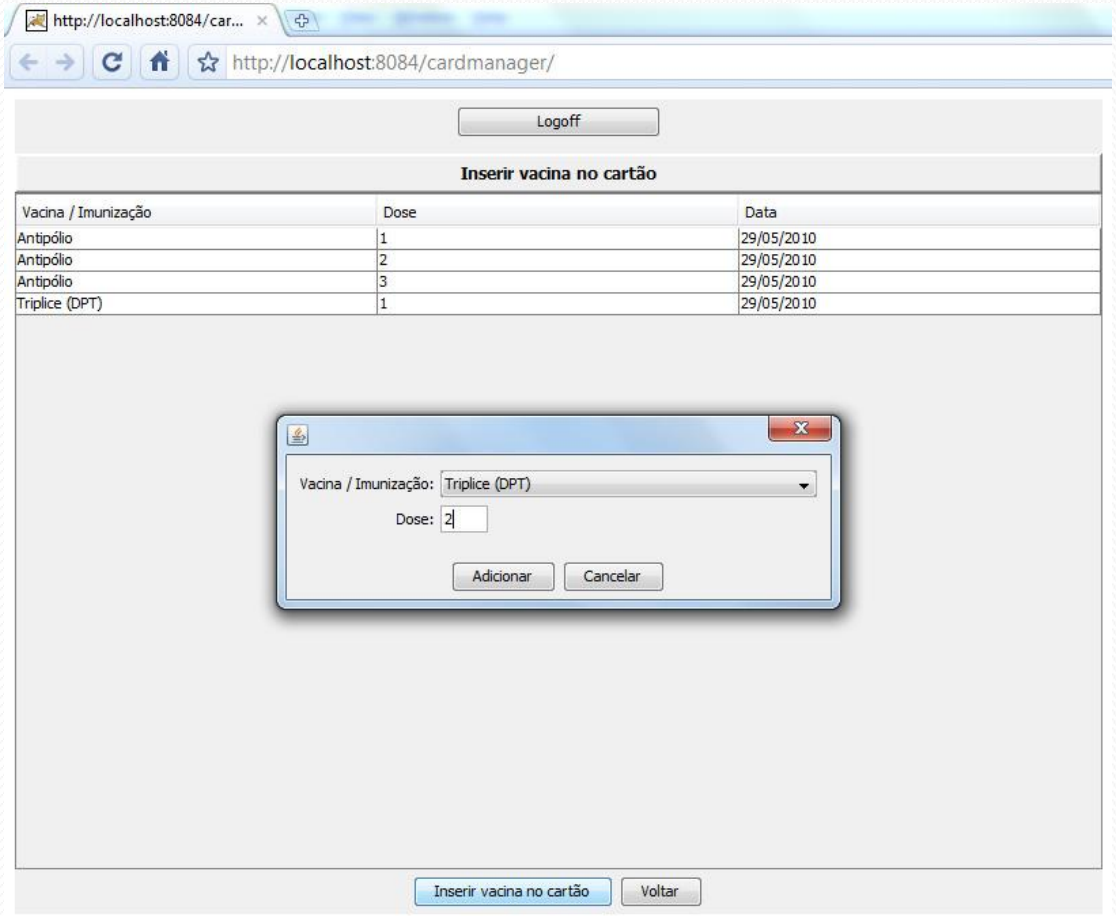
Operacionalidade



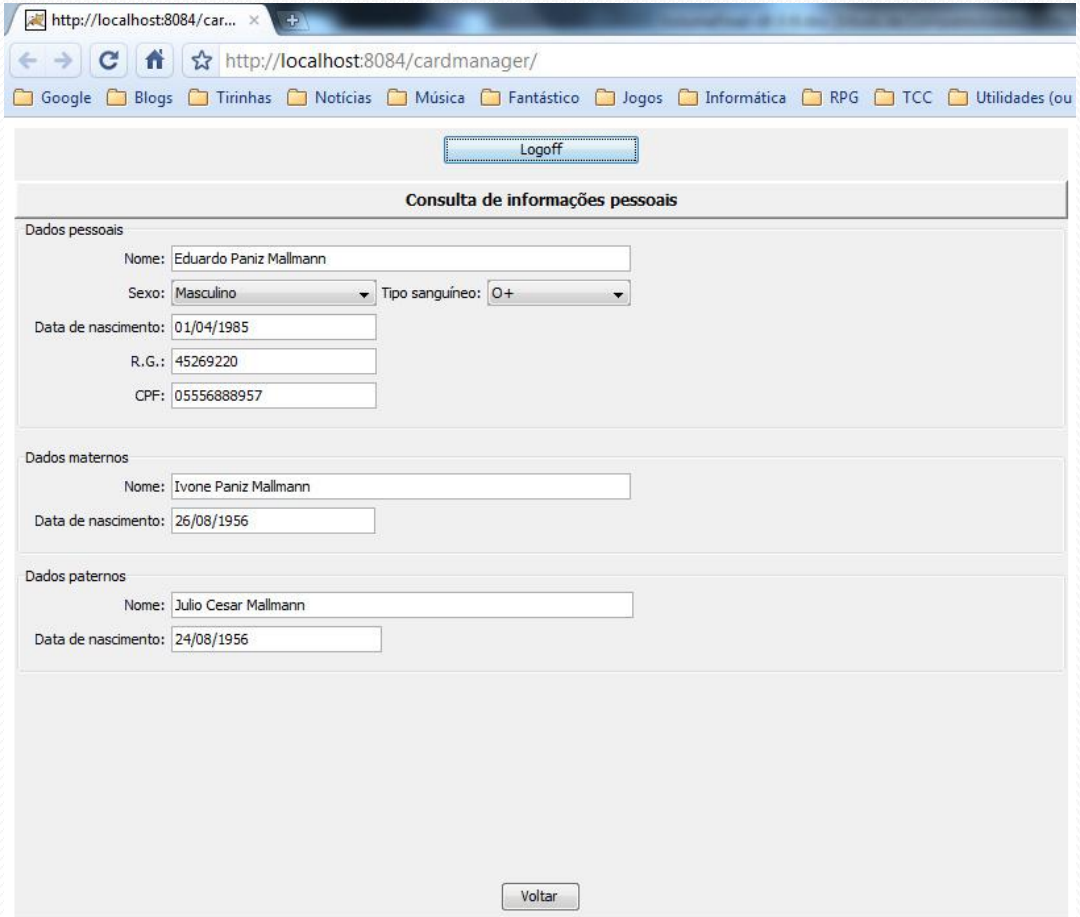
Operacionalidade



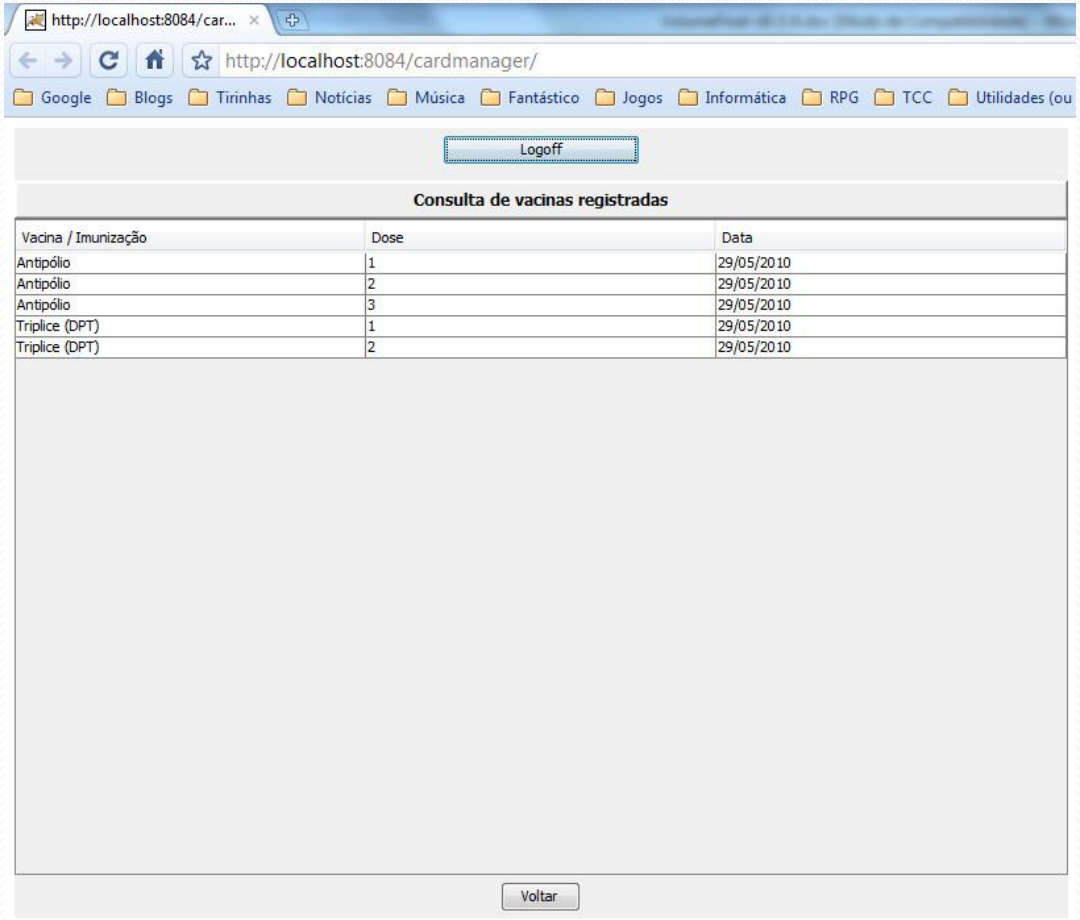
Operacionalidade



Operacionalidade



Operacionalidade



Resultados e discussão

	SUAVI	MINORA; ALEIXO; DIOLINO	MALLMANN
Utilização de smart cards	Sim	Sim	Sim
Persistência de dados no smart card	Não	Não	Sim
Envio de comandos ao smart card	Sim	Sim	Sim
Utilização de smart card pela internet	Não	Não	Sim
Utilização de rotina de criptografia	Sim	Não	Sim
Utilização de certificado digital	Não	Não	Não

Conclusão

- Requisitos propostos foram cumpridos
- Tecnologia Java Card é completa, porém carente de documentação
- Dificuldade de encontrar smart cards no Brasil
- Dificuldade de encontrar documentação completa

Extensões

- Desenvolvimento de toda a caderneta de saúde e unificação de todas
- Testes de durabilidade do cartão e dos dados
- Utilizar última versão da tecnologia Java Card (3.0)
- Utilizar Connected Edition