SOFTWARE PARA VERIFICAÇÃO DE CONFORMIDADE DE SISTEMAS À NORMA ISO/IEC 15408

Dayana Fernanda Trapp
Orientador: Paulo Fernando da Silva

Roteiro

- Introdução
- Objetivos
- Fundamentação teórica
- Desenvolvimento
- Conclusão

Introdução

- Nenhum software é seguro
- Pode-se minimizar os riscos
- Norma ISO/IEC 15408

Fundamentação Teórica

Objetivos

- Disponibilizar um software para analisar um sistema e verificar se o mesmo implementa requisitos de segurança estabelecido pelo norma ISO/IEC 15408.

Objetivos

- Auxiliar um auditor na avaliação
- Analisar um sistema e verificar se ele implementa requisitos da norma ISO/IEC
- Disponibilizar um check-list para fazer a auditoria manual
- Disponibilizar um relatório com os resultados da auditoria

Objetivos

- Disponibilizar um software para analisar um sistema e verificar se o mesmo implementa requisitos de segurança estabelecido pelo norma ISO/IEC 15408.

Segurança da Informação

- Tem como objetivo a proteção da informação para reduzir a probabilidade de incidentes de segurança.
- Princípios básicos
  - Confidencialidade
  - Integridade
  - Disponibilidade

Norma ISO/IEC 15408

- Descreve conceitos necessários para a segurança no desenvolvimento de sistemas

Norma ISO/IEC 15408

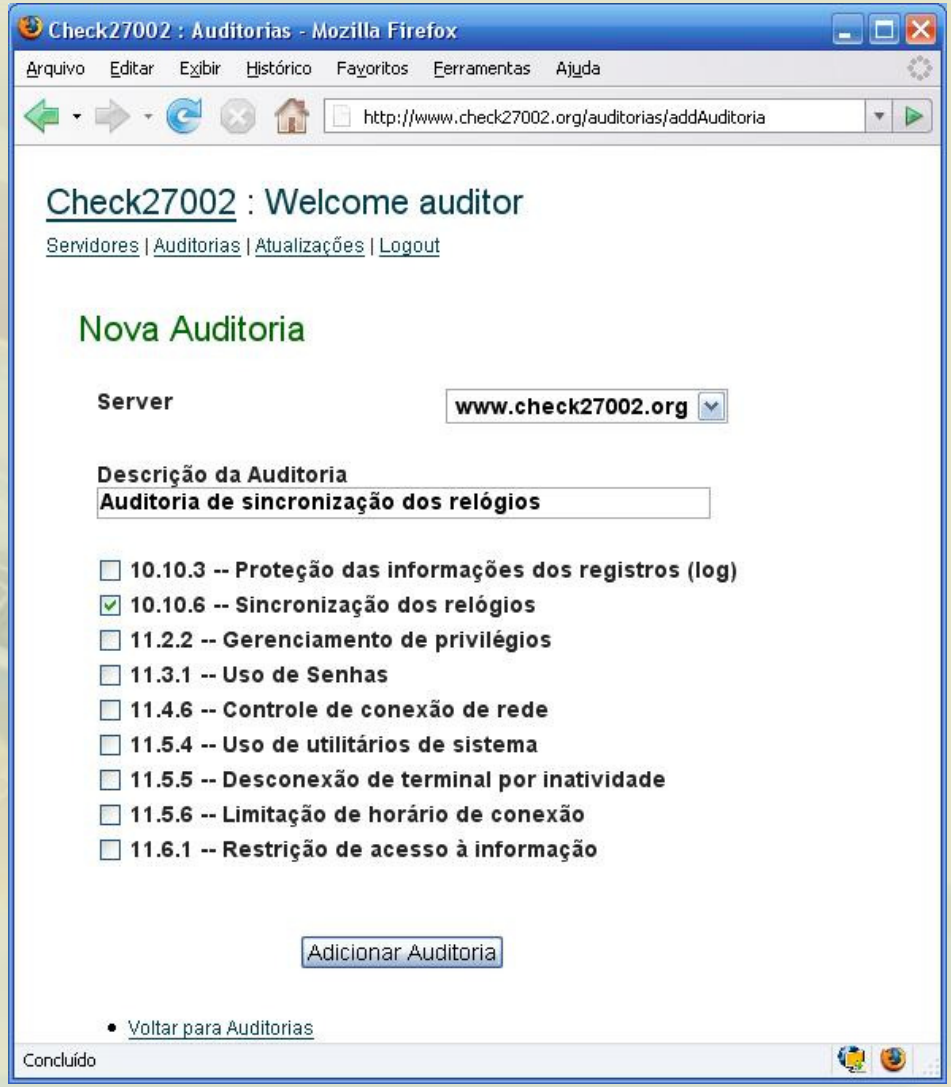
- A norma está dividida em três partes
  - descreve a introdução.
  - cataloga uma série de requisitos funcionais
    - Classe
    - Familia
    - Requisitos
  - Define o critério de avaliação

Auditoria

- Verificar a conformidade com:
  - Objetivos
  - Políticas institucionais
  - Regras
  - Normas

Trabalhos Correlatos

Software para verificação de conformidade de servidores GNU/Linux à norma de segurança NBR ISO/IEC 27002



Software para avaliação da segurança da informação de uma empresa conforme a Norma NBR ISO/IEC 17799

Fortify SCA

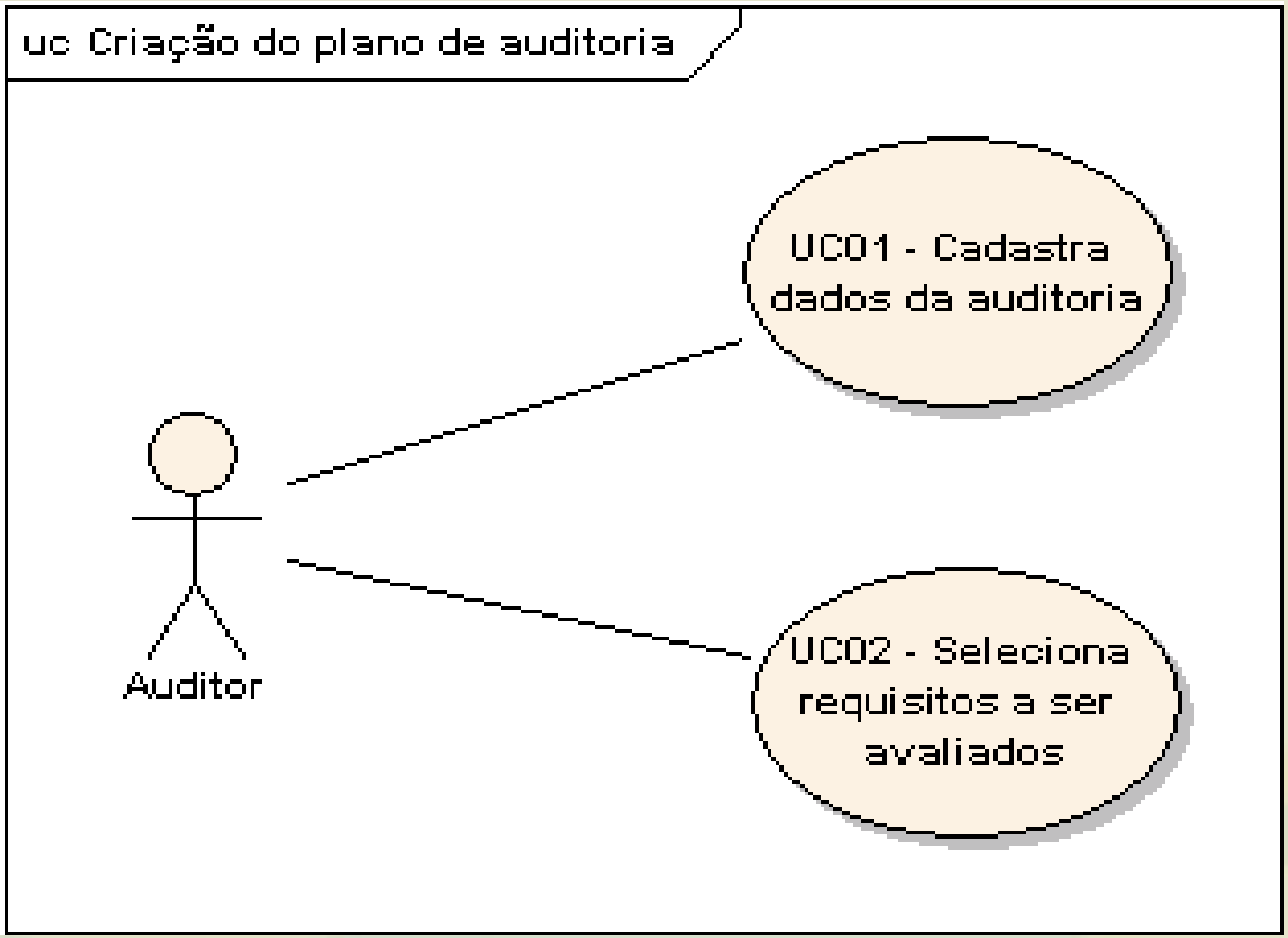
- Verificação de vulnerabilidades
- Análise estática
- Regras próprias

Desenvolvimento

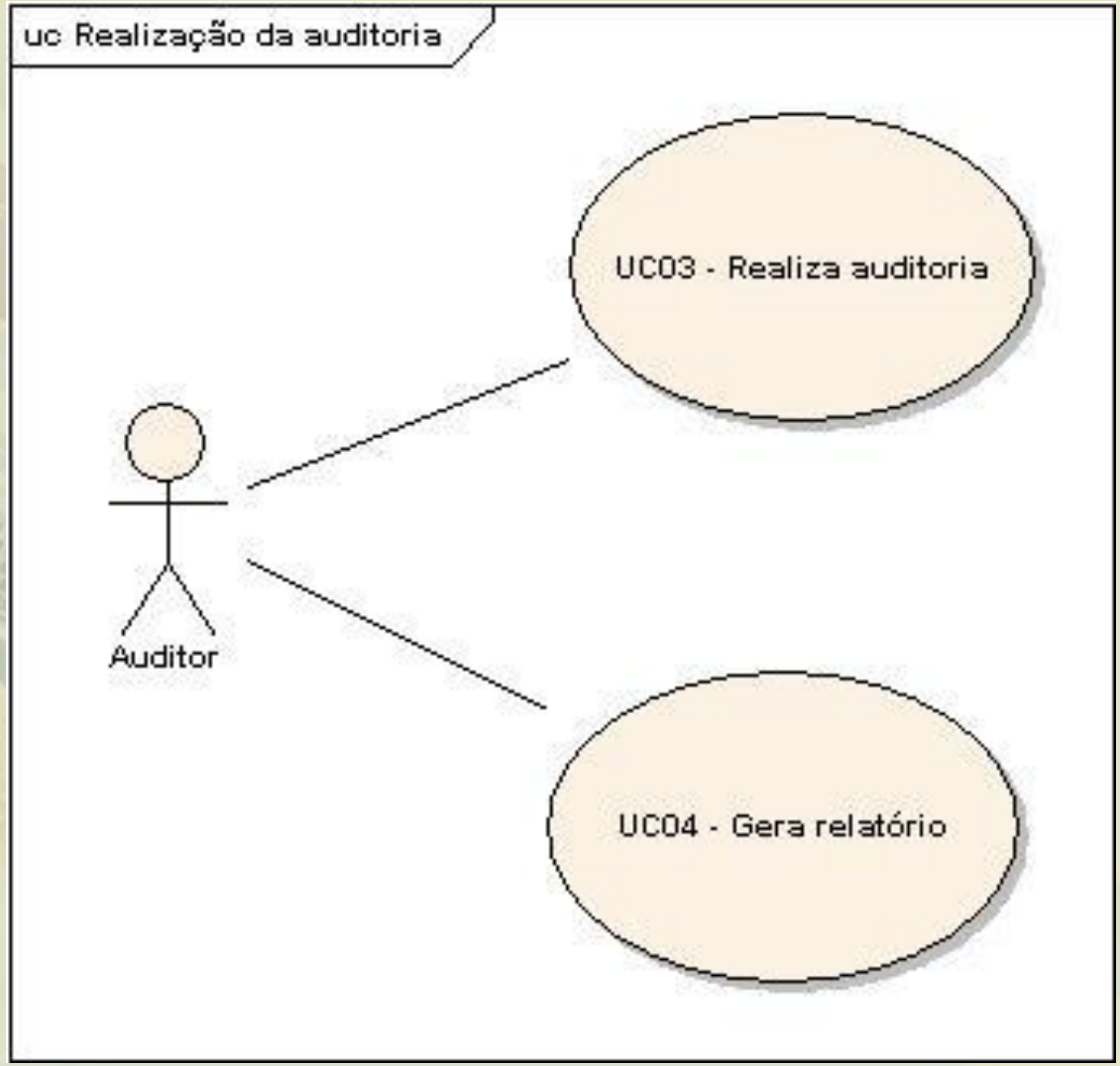
Requisitos do Sistema

- Fornecer uma lista de requisitos
- Adicionar *plugins* para futuras extensões
- Fornecer uma biblioteca de extensão
- Exibir uma lista de *check-list* para fazer a auditoria
- A auditoria deve ser feita em sistemas desenvolvidos em Java
- Gerar relatório

Caso de Uso


uc Criação do plano de auditoria
UC01 - Cadastra dados da auditoria
Auditor
UC02 - Seleciona requisitos a ser avaliados

Caso de Uso


uc Realização da auditoria
UC03 - Realiza auditoria
Auditor
UC04 - Gera relatório

Arquitetura do Sistema

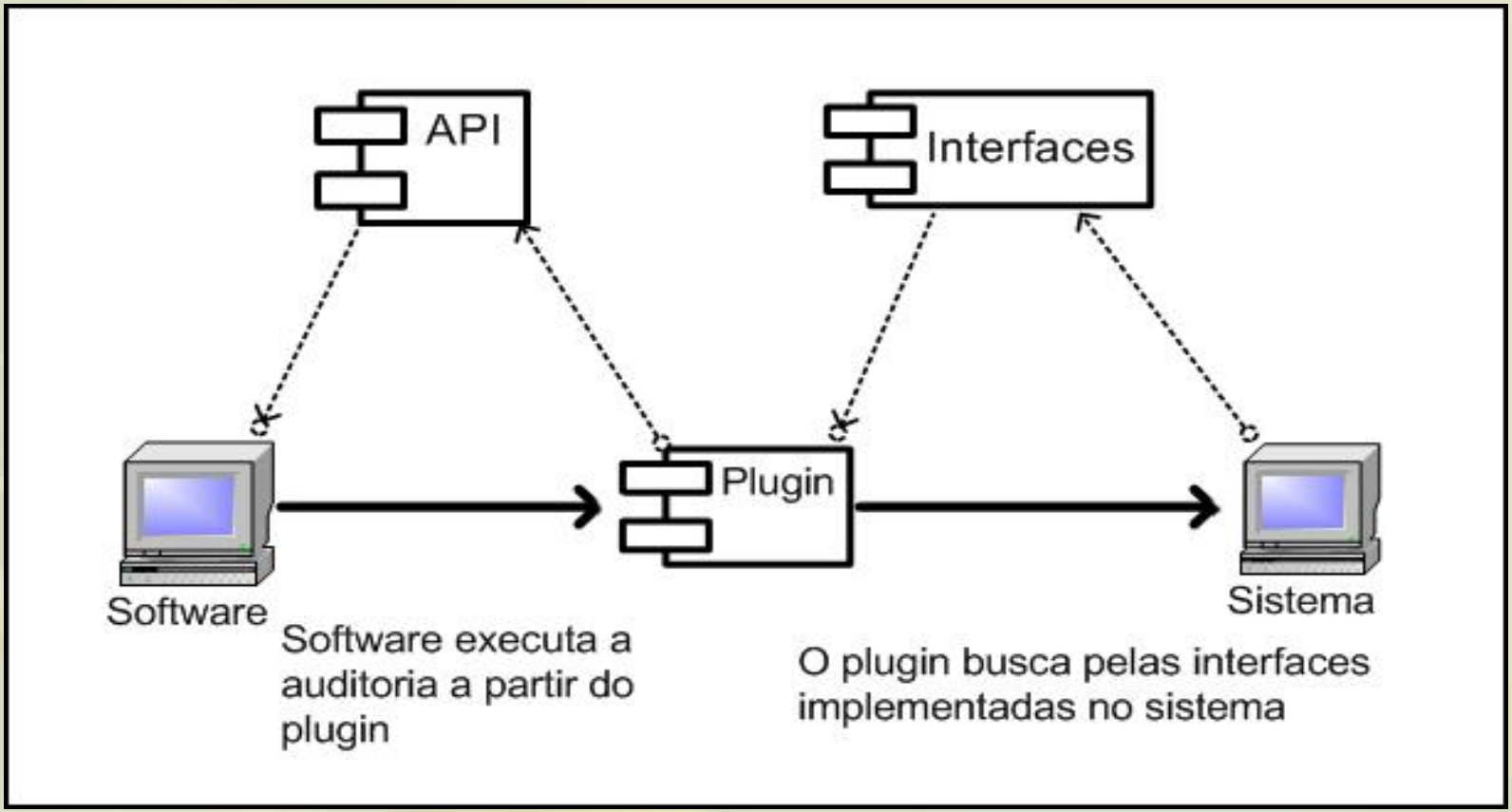


Diagrama de Classe Software

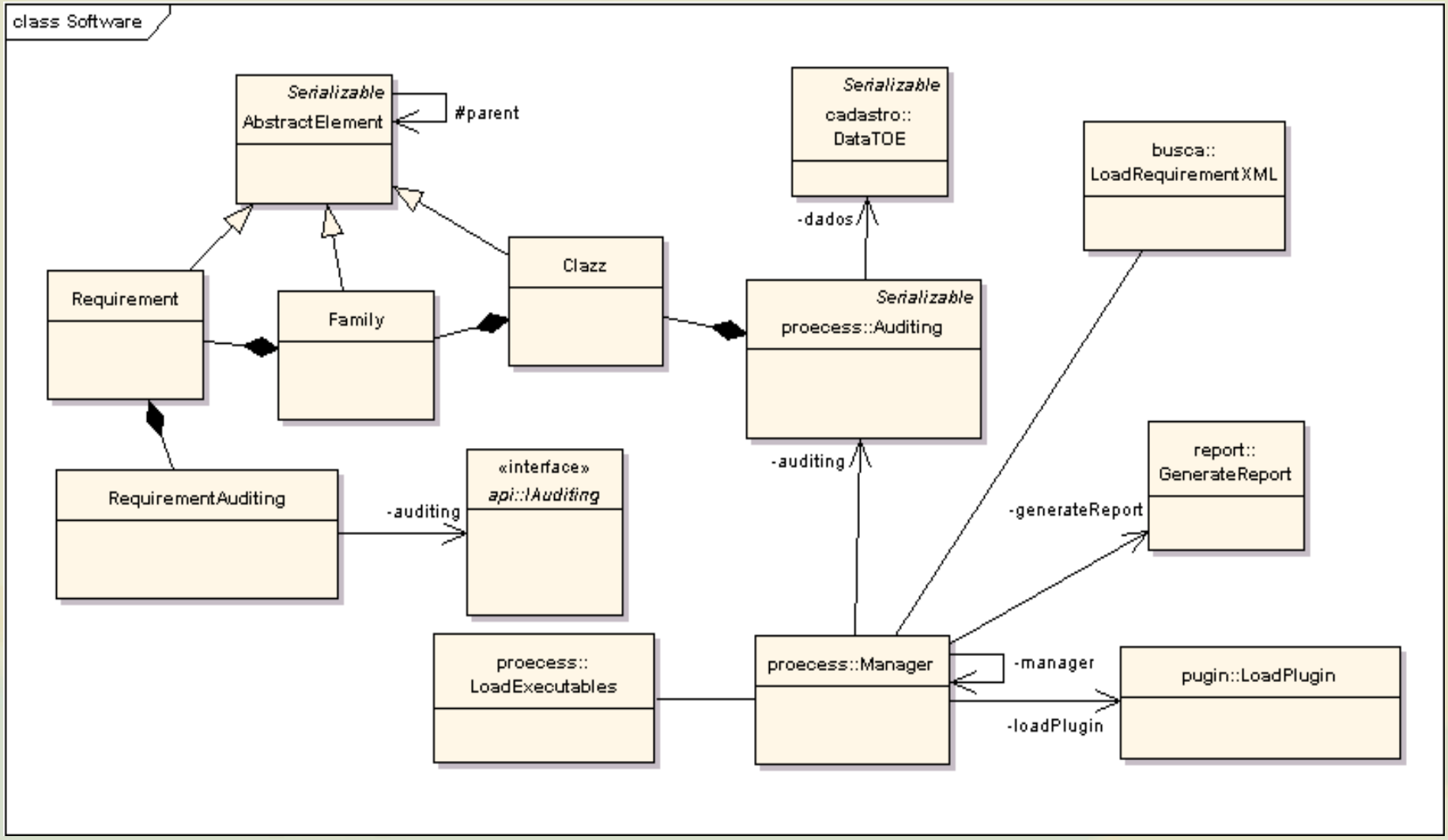


Diagrama de Classe
API

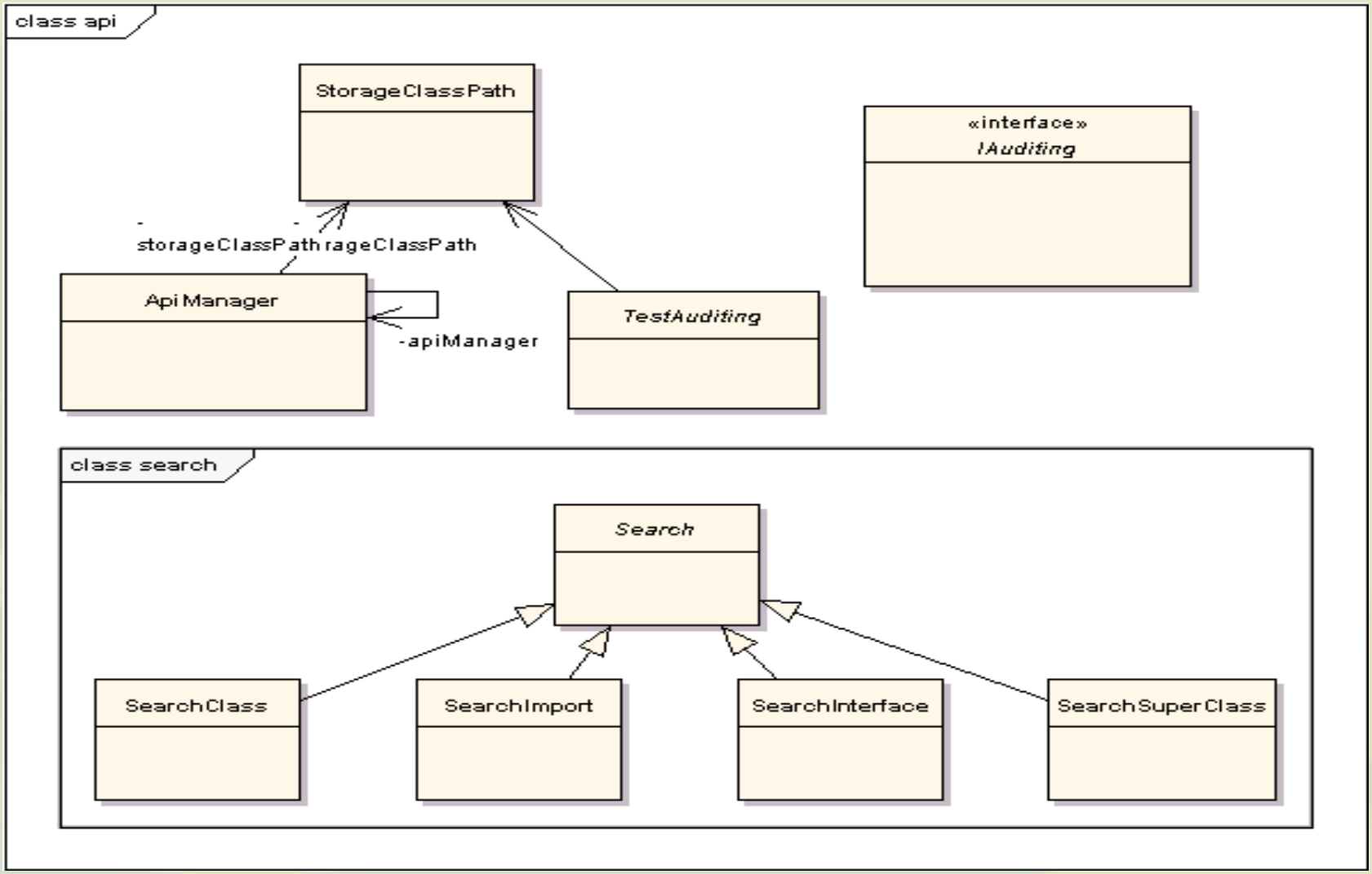


Diagrama de Classe Plugin

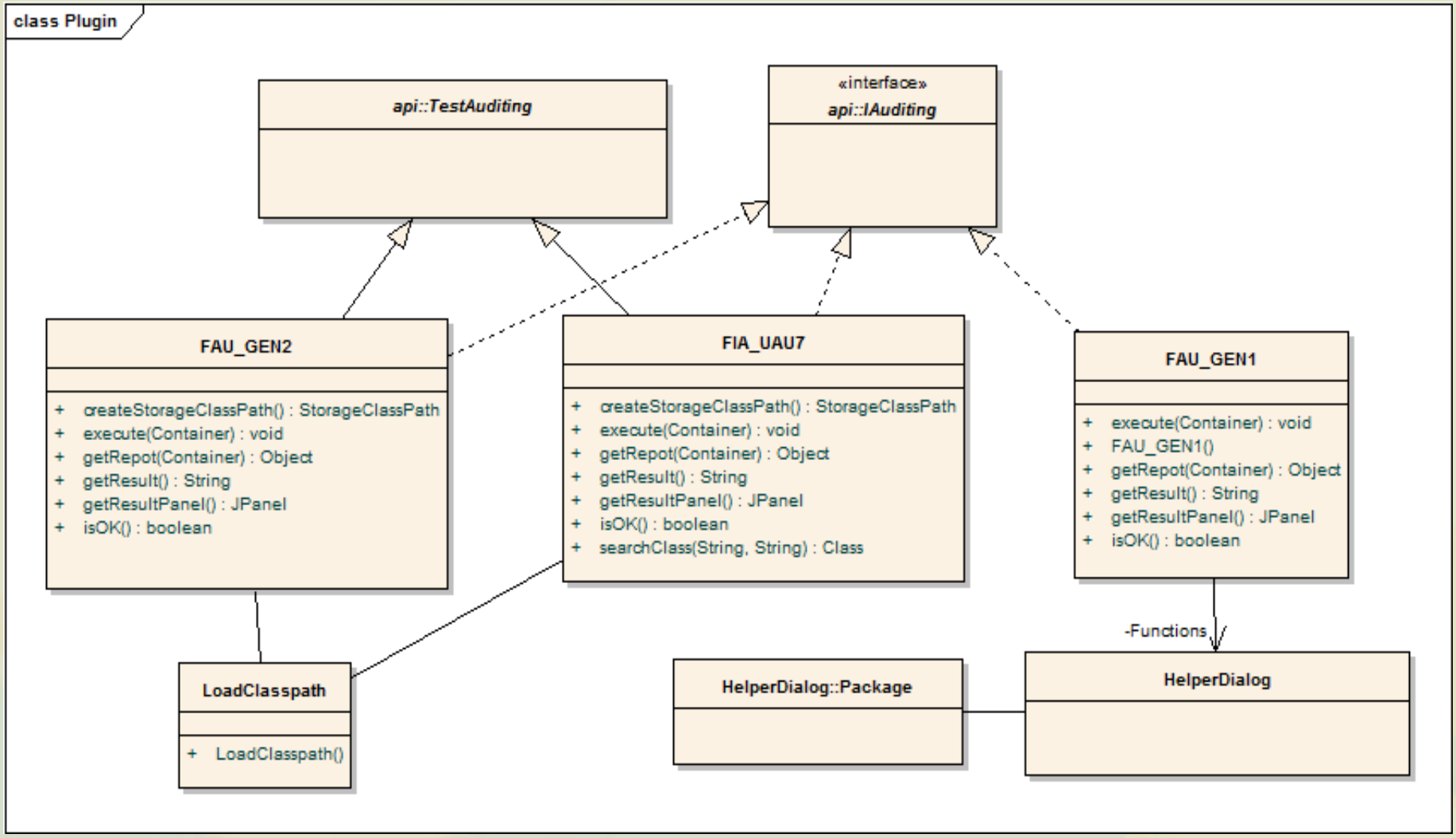
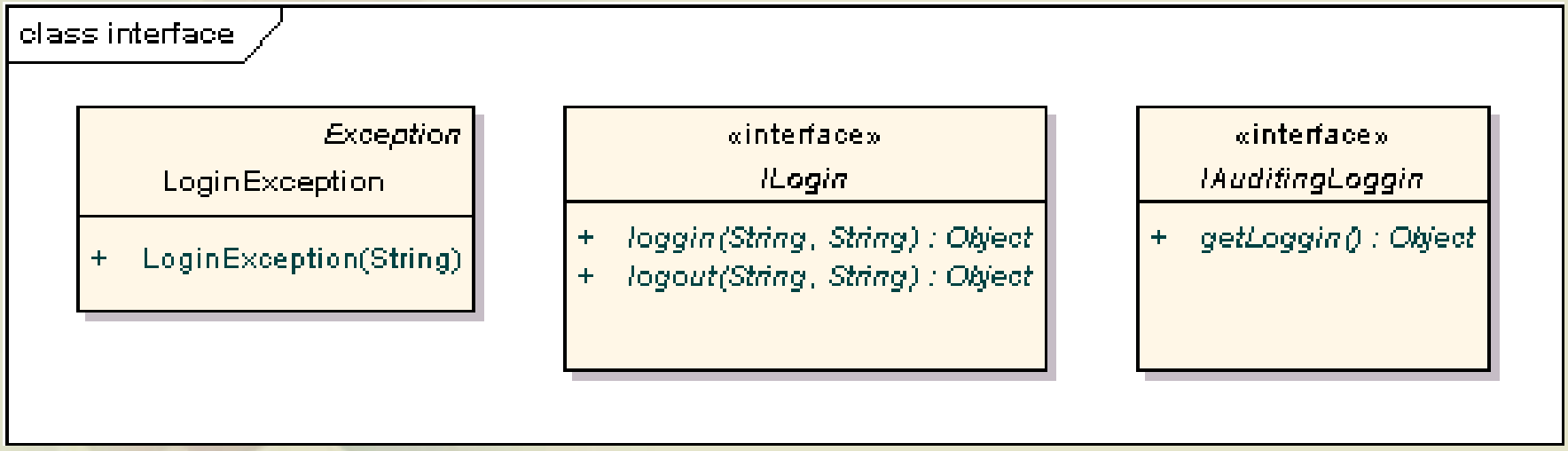


Diagrama de Classe Interfaces



Implementação

Estudo de Caso

- Sistema analisado é o Scrump
- Requisitos da norma que possui teste automatizado
  - FAU_GEN.1
  - FAU_GEN.2
  - FIA_UAU.7

Ferramentas utilizadas

- Dom4j
- Qdox
- iText

Dom4j
Estrutura XML

```
<classe id="FAU" descricao="AUDITORIA DE SEGURANÇA">
  <familia id="FAU_ARP"
           descricao="FAU_ARP ? Resposta automática a
    auditoria">
    <requisito id="FAU_ARP.">
       FAU_ARP.1 - Alarmes de segurança
    </requisito>
   </familia>
</classe>
```

Dom4j

```
public class LoadRequirementXML {
     ArrayList<Clazz> listClazz = new ArrayList<Clazz>();
    /*Itera por todos os nós do XML e cria um novo objeto de acordo com o atributo lido*/
    private void createTreeElement(Element element) {
        for (Iterator<Element> iterator = element.elementIterator(); iterator.hasNext();) {
            Element classElement = iterator.next();

            if (classElement.getName().equals("classe")) {
                Clazz clazz = createClazz(classElement);
                listClazz.add(clazz);

                for (Iterator<Element> iteFam = classElement.elementIterator();
                iteFam.hasNext();) {
                    Element famElement = iteFam.next();
                    Family family = createFamily(famElement, clazz);
                    clazz.addFamilias(family);

                    for (Iterator<Element> iteReq = famElement.elementIterator();
                    iteReq.hasNext();) {
                        Element reqElement = iteReq.next();
                        Requirement requirement =  createRequirement(reqElement, family);
                        family.addRequisito(requirement);
                    }
                }
            }
        }
    }
}
```

Qdox

```
protected void search(File file) {
        if (!file.getName().contains(JAVA_SOURCE)) {
            return;
        }
        try {
            JavaDocBuilder n = new JavaDocBuilder();
            n.addSource(file);
            JavaSource javaSource = n.getSources()[0];
            for (JavaClass jClass : javaSource.getClasses()) {
                inter: for (JavaClass interClass :
                    jClass.getImplementedInterfaces()) {
                    for (String find : interfaceNames) {
                        if (interClass.getName().equals(find)) {
                            getCLassFound().add(file);
                            break inter;
                        }
                    }
                }
            }
        } catch (FileNotFoundException e) {
            throw new RuntimeException(e);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
```

iTEXT

```
public Document createDocument(String path)
    throws FileNotFoundException, DocumentException {
    Document document = new Document(PageSize.A4, 50, 50, 50, 50);
    //cria o documento
    PdfWriter.getInstance(document, new FileOutputStream(path));
    document.open();
    return document;
}
public void writeClazz(String path, Auditing auditing)
    throws FileNotFoundException, DocumentException {

        Document document;
        createFontStyle();
        document = createDocument(path);
        createDataTOE(document, auditing.getDados());
        createParagraphClazz(document, auditing.getListClazz());

    }
```

API

```
public interface IAuditing {

    void execute(Container container);

    boolean isOK();

    Container getResultPanel();

    String getResult();

    Object getReport();
}
```

FAU_GEN.1

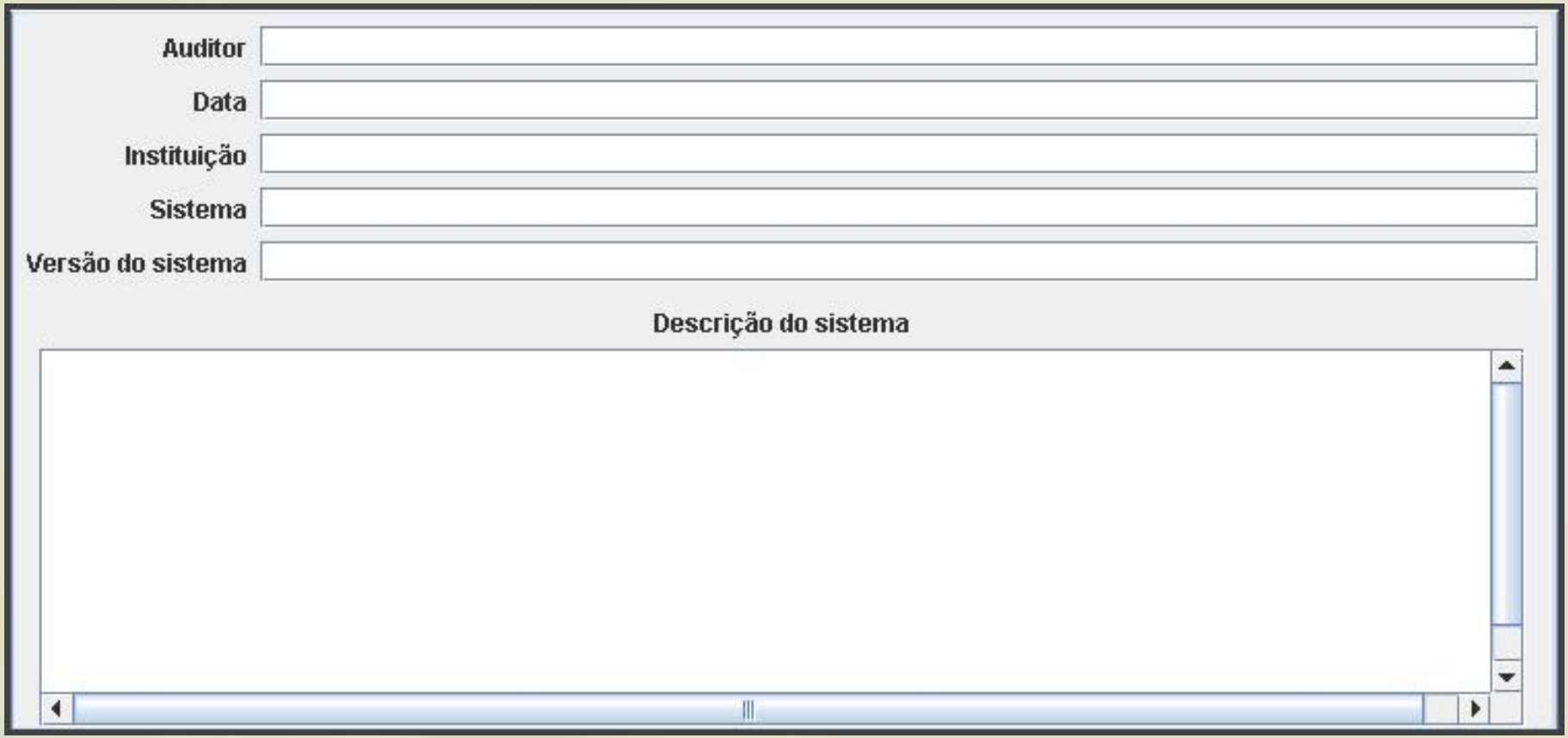
```
public SearchImport searchImport(String importLogger, String directory) {

        SearchImport searchClass = new SearchImport(importLogger);
        searchClass.search(directory);

         if (searchClass.getCLassFound().size() == 0) {
                 JOptionPane.showMessageDialog(null, "Não encontrou " +
                 "nenhuma classe com o import " + importLogger);
                 }

                 return searchClass;
         }
```

FIA_UAU.7

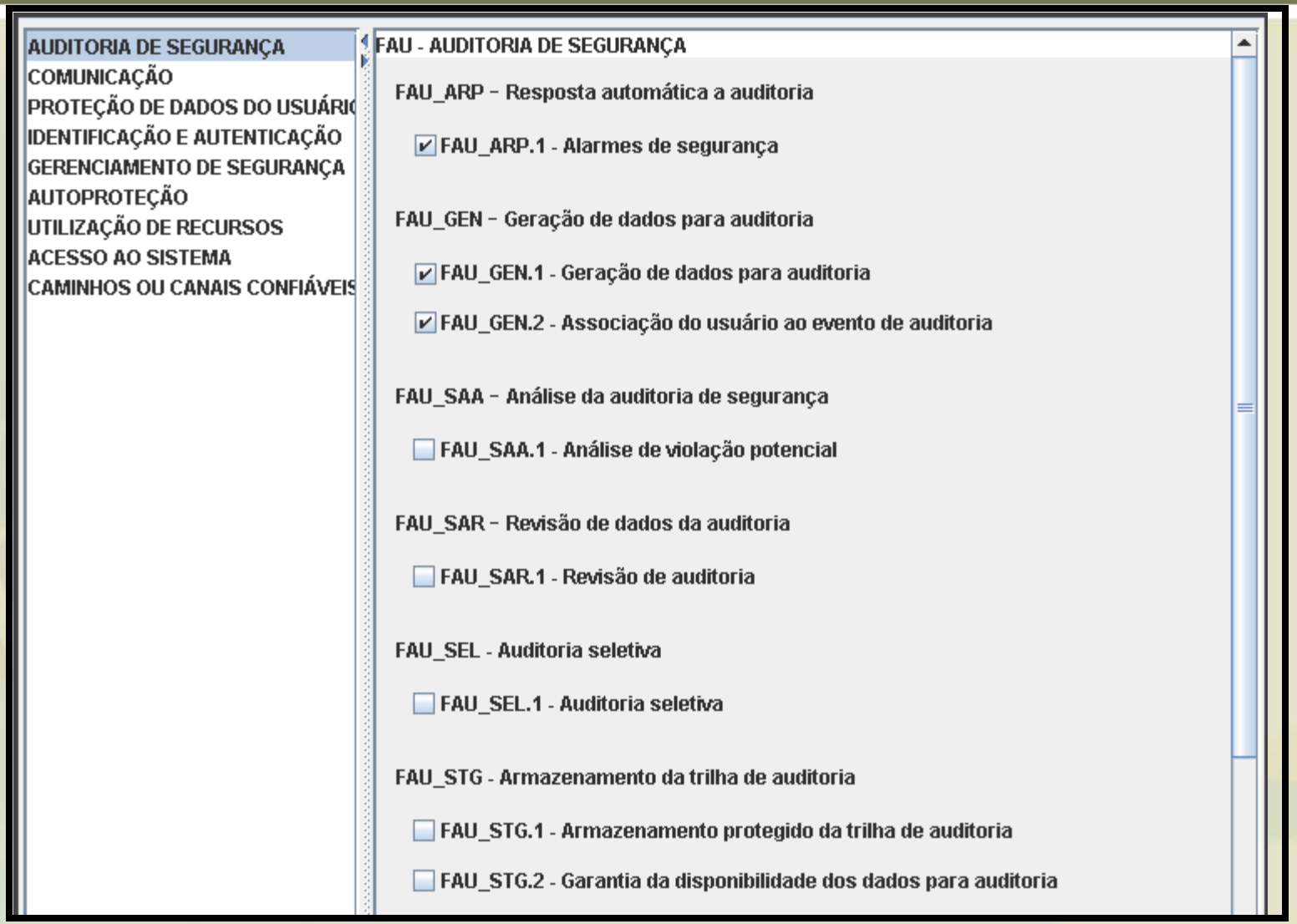
```
private void testAuditingLoginWrong(String login,String pass) throws ClassNotFoundException,
                            SecurityException, NoSuchMethodException {
        result.append("Mensagem de erro deve ser genérica: \n");
        String msg = "";
        Class<?> forName = searchClass("furb.br.seguranca.ILogin");
        if (forName == null) {
        JOptionPane.showMessageDialog(null, "Não foi encontrada nenhuma classe " +
                                            "para a realização da auditoria");
        }
        Method method = forName.getMethod("loggin",new Class[] {String.class,String.class});
        try {
                  Object invoke = method.invoke(forName.newInstance(), login, pass);
                  results.add(false);
                  result.append("Não deveria ter logado\n");

         } catch (IllegalArgumentException e) {
                    msg = e.getCause().getMessage();
        } catch (IllegalAccessException e) {
                  msg = e.getCause().getMessage();
        } catch (InvocationTargetException e) {
                  // Adiciona a msg do erro do login
                  msg = e.getCause().getMessage();
        } catch (InstantiationException e) {
                  msg = e.getCause().getMessage();
        }
        result.append("\n" + msg);
        msgVerify(msg); // ok

}
```

Operacionalidade

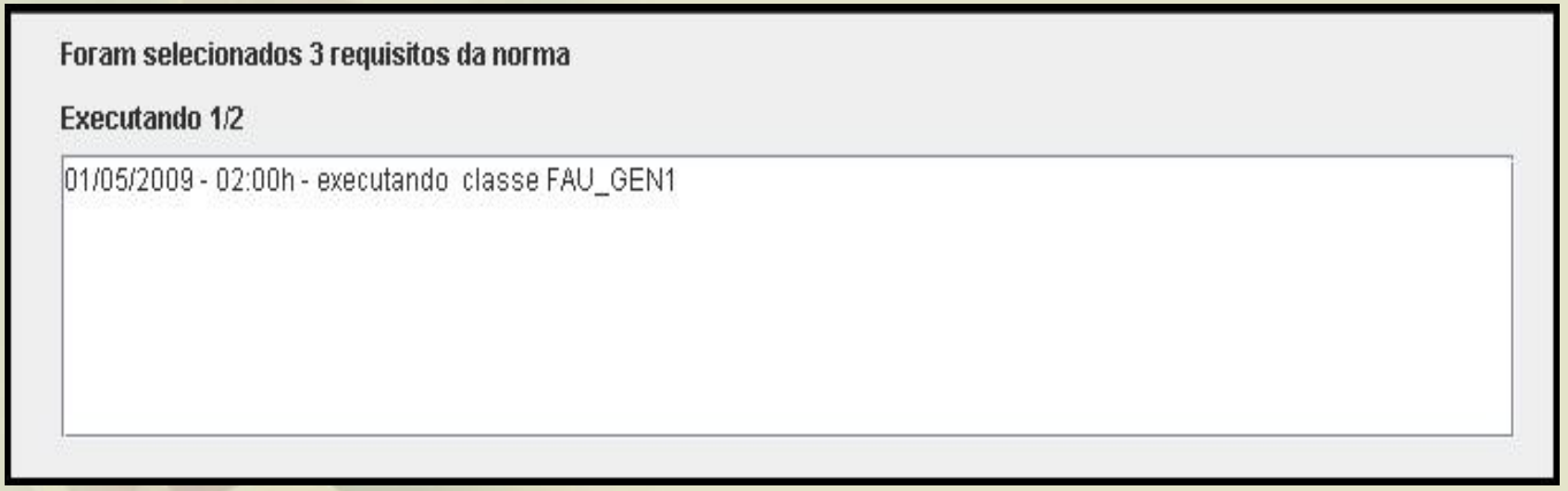
Cadastro


Auditor
Data
Instituição
Sistema
Versão do sistema
Descrição do sistema

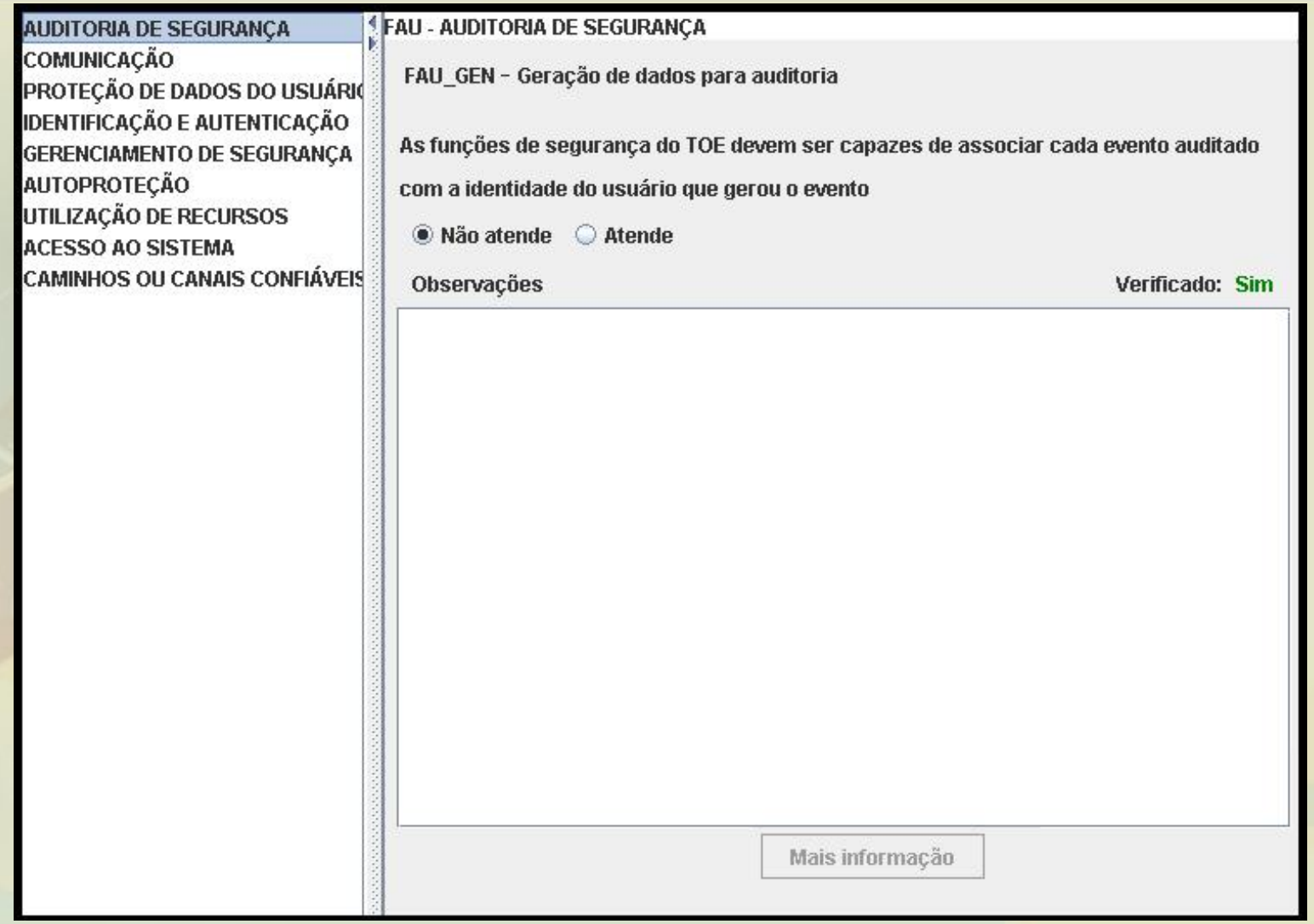
Requisitos


AUDITORIA DE SEGURANÇA
COMUNICAÇÃO
PROTEÇÃO DE DADOS DO USUÁRIO
IDENTIFICAÇÃO E AUTENTICAÇÃO
GERENCIAMENTO DE SEGURANÇA
AUTOPROTEÇÃO
UTILIZAÇÃO DE RECURSOS
ACESSO AO SISTEMA
CAMINHOS OU CANAIS CONFIÁVEIS
FAU - AUDITORIA DE SEGURANÇA
FAU_ARP - Resposta automática a auditoria
FAU_ARP.1 - Alarmes de segurança
FAU_GEN - Geração de dados para auditoria
FAU_GEN.1 - Geração de dados para auditoria
FAU_GEN.2 - Associação do usuário ao evento de auditoria
FAU_SAA - Análise da auditoria de segurança
FAU_SAA.1 - Análise de violação potencial
FAU_SAR - Revisão de dados da auditoria
FAU_SAR.1 - Revisão de auditoria
FAU_SEL - Auditoria seletiva
FAU_SEL.1 - Auditoria seletiva
FAU_STG - Armazenamento da trilha de auditoria
FAU_STG.1 - Armazenamento protegido da trilha de auditoria
FAU_STG.2 - Garantia da disponibilidade dos dados para auditoria

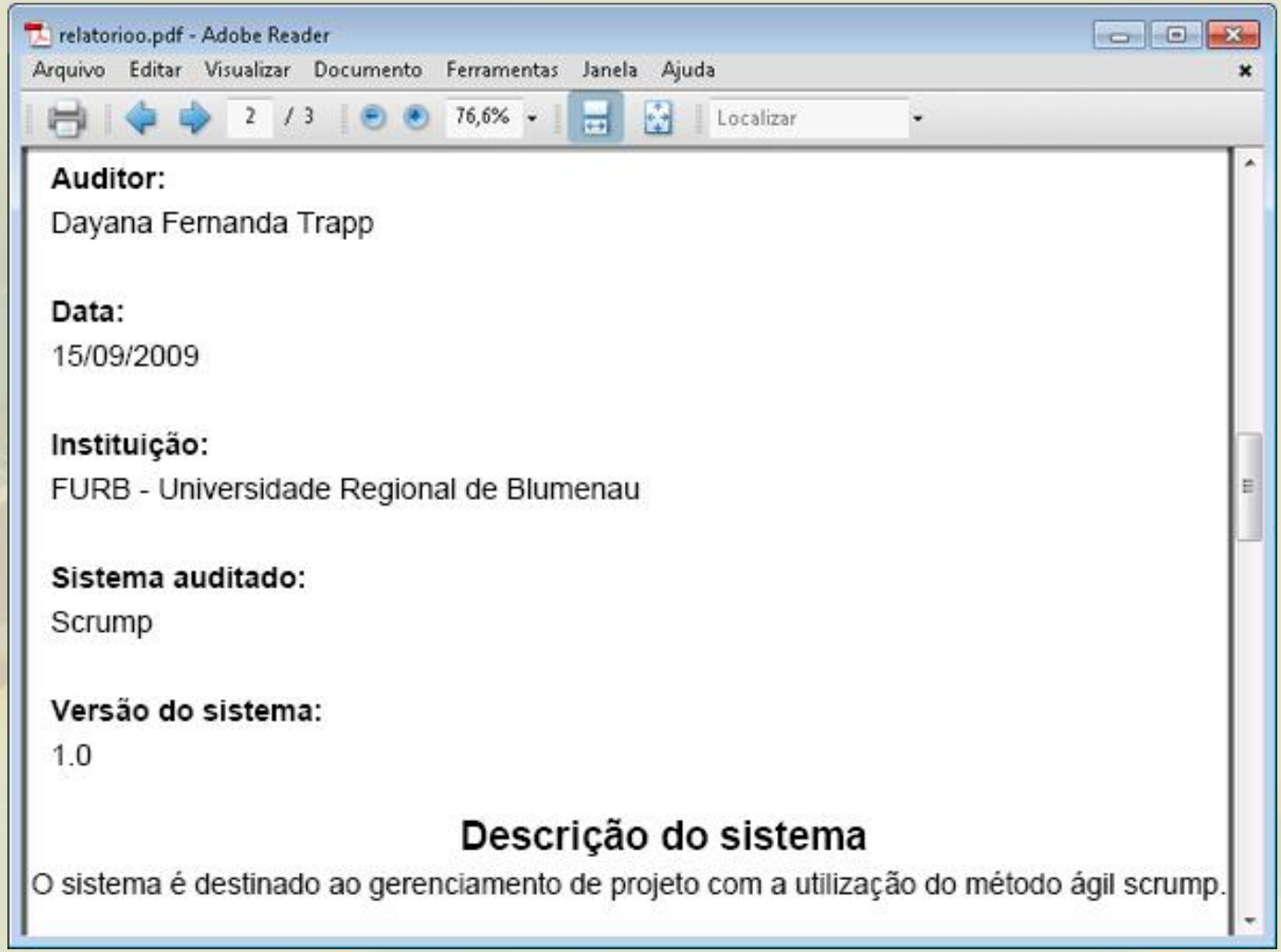
Status


Foram selecionados 3 requisitos da norma
Executando 1/2
01/05/2009 - 02:00h - executando classe FAU_GEN1

Resultado


AUDITORIA DE SEGURANÇA
COMUNICAÇÃO
PROTEÇÃO DE DADOS DO USUÁRIO
IDENTIFICAÇÃO E AUTENTICAÇÃO
GERENCIAMENTO DE SEGURANÇA
AUTOPROTEÇÃO
UTILIZAÇÃO DE RECURSOS
ACESSO AO SISTEMA
CAMINHOS OU CANAIS CONFIÁVEIS
FAU - AUDITORIA DE SEGURANÇA
FAU_GEN – Geração de dados para auditoria
As funções de segurança do TOE devem ser capazes de associar cada evento auditado com a identidade do usuário que gerou o evento
Não atende
Atende
Observações
Verificado: Sim
Mais informação

Relatório


relatorioo.pdf - Adobe Reader

Arquivo Editar Visualizar Documento Ferramentas Janela Ajuda

2 / 3 76,6% Localizar

Auditor:
Dayana Fernanda Trapp

Data:
15/09/2009

Instituição:
FURB - Universidade Regional de Blumenau

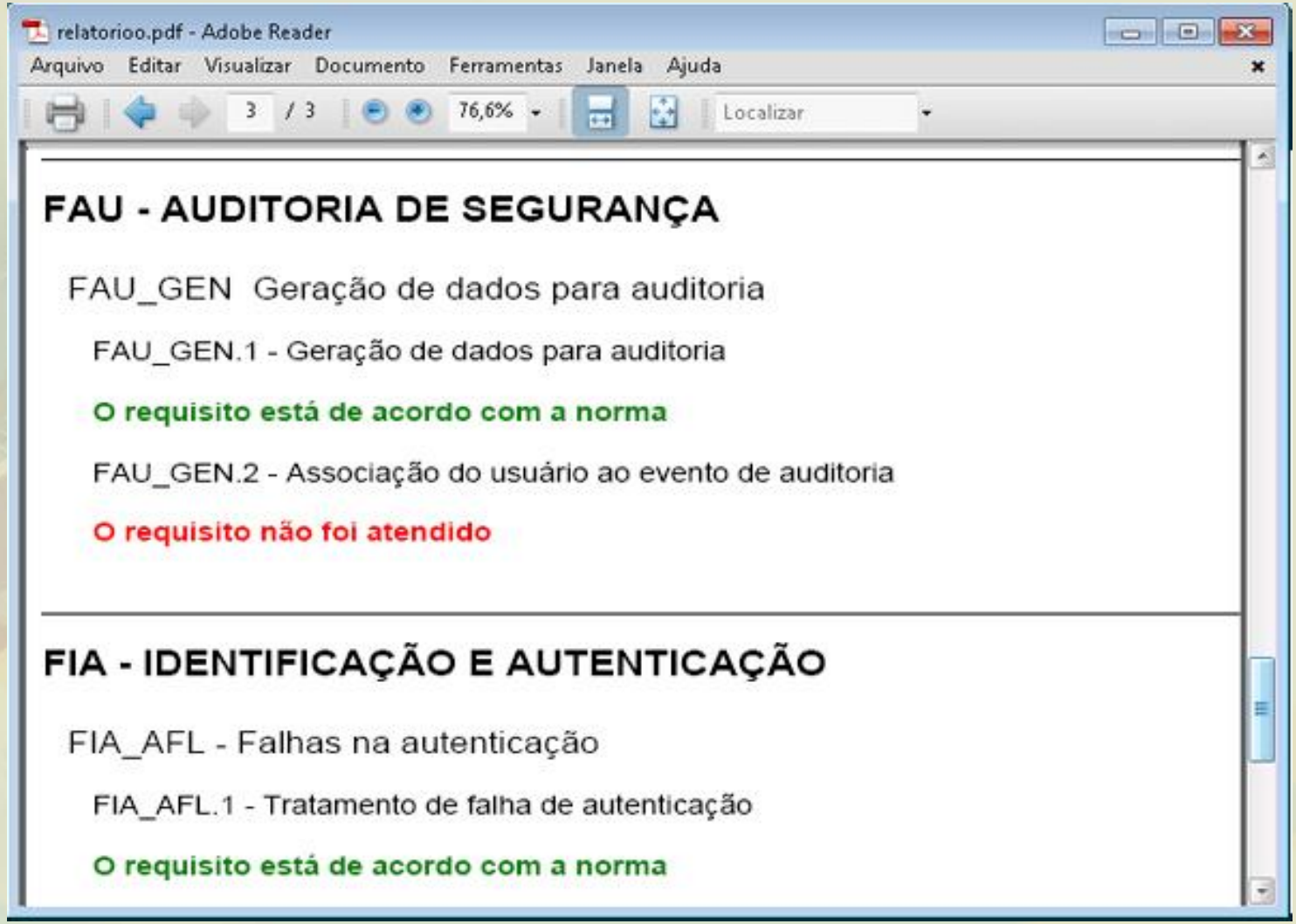
Sistema auditado:
Scrump

Versão do sistema:
1.0

Descrição do sistema

O sistema é destinado ao gerenciamento de projeto com a utilização do método ágil scrump.

Relatório


relatorioo.pdf - Adobe Reader
Arquivo Editar Visualizar Documento Ferramentas Janela Ajuda
3 / 3
76,6%
Localizar
FAU - AUDITORIA DE SEGURANÇA
FAU_GEN Geração de dados para auditoria
FAU_GEN.1 - Geração de dados para auditoria
O requisito está de acordo com a norma
FAU_GEN.2 - Associação do usuário ao evento de auditoria
O requisito não foi atendido
FIA - IDENTIFICAÇÃO E AUTENTICAÇÃO
FIA_AFL - Falhas na autenticação
FIA_AFL.1 - Tratamento de falha de autenticação
O requisito está de acordo com a norma

Finalizando....

Resultado e Discussão

- Dificuldade encontrar uma regra
- Vantagens dos plugins
  - Implementar diversas formas de testes
  - Desenvolver testes para um determinado sistema
  - Desenvolver testes para outras linguagens
  - Utilizar outros tipos de testes

Conclusão

- O trabalho teve seu objetivo alcançado
- Segurança é importante
- Preocupação do desenvolvimento
- Regras definidas pela norma 15408
- O software desenvolvido auxilia um auditor na auditoria de um sistema

Extensões

- Implementar mais testes automatizados
- Implementar *plugins* para outras linguagens
- Melhorar o relatório