Universidade de Coimbra
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Informática
Mestrado em Engenharia Biomédica

Informática Médica

Trabalho Prático 5

DICOM

Elaborado por:

Edite Figueiras
João Duarte
Ricardo Martins

Coimbra, 19 de Dezembro de 2007

I. OBJECTIVO

Através da realização deste trabalho prático pretende-se desenvolver uma aplicação que permita ao utilizador especificar a directoria de um DICOMDIR e baseando-se nessa indicação, listar os diversos exames do tipo imagem pertencentes à DICOMDIR seleccionada. Pretende-se também que ao seleccionar um dos exames listados, sejam exibidos os atributos da *Directory Record* do tipo *Image* correspondente ao exame seleccionado pelo utilizador.

Como objectivo adicional, pretendeu-se desenvolver um visualizador da sequência de imagens do exame seleccionado.

II. IMPLEMENTAÇÃO

- Software utilizado

O trabalho foi desenvolvido na linguagem de programação *Java*, tendo-se recorrido ao ambiente de desenvolvimento *NetBeans IDE 5.5.1* associado ao *jdk 1.6._0.03*. Recorreu-se, como suporte ao desenvolvimento e teste do trabalho, ao código base e ao DICOMDIR fornecidos.

Utilizaram-se as seguintes bibliotecas:

- Dicom Image I/O plug-in 1.10 (ficheiro *dicom.jar*)
  Permite a leitura e manipulação de ficheiros Dicom.

- Java Image I/O tools 1.0.01 (ficheiro *jai_imageio.jar*)
  Possibilita a exibição de imagens Dicom que foram comprimidas utilizando, entre outros tipos possíveis, o tipo de compressão JPEG 2000.

- Swing-layout (ficheiro *swing-layout-1.0.jar*)
  Necessário ao desenvolvimento de interfaces gráficos potáveis entre diversas plataformas.

III. CLASSES E MÉTODOS UTILIZADOS

De forma a implementar uma aplicação com as funcionalidades descritas no objectivo deste trabalho, utilizaram-se as classes que se apresentam de seguida. Para cada uma das classes principais, descrevem-se os métodos mais importantes.

- **class ExemploDicomDir**

É a classe principal do projecto. É aqui que se implementa o interface gráfico da aplicação. O interface gráfico desenvolvido tem cinco componentes fundamentais:

- JTextField txtPath

Caixa de texto destinada à introdução do endereço da directoria onde se encontra o DICOMDIR.

- JButton jButton1

Botão que exibe o texto "Show". A sua função é descrita numa fase posterior do trabalho.

- JTable tableExames

Tabela onde são exibidos em cada linha, dados referentes a cada um dos exames do tipo imagem relacionados com a DICOMDIR.

- JTextArea txtArea

Caixa de texto onde são exibidos os atributos do *Directory Record* do tipo *Image* correspondentes ao exame seleccionado.

- JButton btn_mostrarImagens

Botão com o texto "Mostrar Imagens". A sua função é descrita numa fase posterior do trabalho.

Os métodos mais relevantes que se utilizam nesta classe são:

- void initComponents()

É neste método que são inicializados e destribuídas todas as componentes (botões, caixas de texto,…) que constituem o interface gráfico definido. Relativamente ao código original fornecido, adicionou-se um novo botão (btn_mostrarImagens), alterou-se a dimensão da *label* próxima do *txtPath* e ainda o título exibido na janela principal da aplicação.

- void jButton1ActionPerformed(ActionEvent evt)

Método que é executado sempre que se clica no botão jButton1. Ao ser executado, este método invoca o método estático leDirectorio da classe ReadDicomDir, o qual devolve um vector que contém vários tipos de dados. Deste são extraídos os dados a serem exibidos na tabela tableExames, assim como, os dados referentes aos atributos das Patient Directory Record, Study Directory Record, Series Directory Record, Image Directory Record, que foram necessárias percorrer até chegar-se à Image Directory Record de cada um dos exames.

São exibidos em cada entrada da tabela tableExames, um conjunto de dados (ID paciente, Nome Paciente, Data Nascimento, Tipo Exame) descritivo de cada exame.

- void valueChanged(ListSelectionEvent e)

Este método é invocado sempre que a linha seleccionada na tabela tableExames muda. O código deste método foi modificado relativamente a aquele que foi fornecido. Verificou-se durante a implementação e teste da aplicação que o método original não identificava correctamente o índice da linha seleccionada na tabela.

O código que se implementou obtém o índice da linha da tabela seleccionada, extrai o objecto que contém o conjunto de atributos do exame referente à linha seleccionada na tabela e exibe em txtArea os atributos da Image Directory Record do exame seleccionado.

- void btn_mostrarImagensActionPerformed(ActionEvent evt)

Método que é executado sempre que é clicado o botão btn_MostrarImagens.

Caso esteja seleccionado um dos exames listados na tabela tableExames, procede-se à extracção do valor da directoria relativa às imagens do exame seleccionado e do valor dos atributos PatientID, PatientsName, Modality, NumberOfFrames, FrameTime. Estes dados são passados como argumentos ao construtor da classe visualizadorAnimacao.

Caso não esteja seleccionado algum exame na tabela é exibida uma mensagem de alerta que indica ao utilizador a obrigatoriedade da selecção de um exame antes de pretender visualizar a sequência de imagens que o compõe.

class Atributes

Esta classe foi definida para guardar os atributos das Directory Record do tipo Patient, Study, Series e Image de cada um dos exames listados na tabela tableExames.

class ReadDicomDir

Esta classe implementa o seguinte método estático.

- Vector leDirectorio(String path)

Este método devolve um vector, resultadosLeitura, com cinco elementos, sendo cada um deles um vector: dadosTabela, atributosExames, filesExames, frameTime e numFrames.

Dado que este método implementa uma funcionalidade fundamental ao desenvolvimento da aplicação, a sua implementação é descrita com maior pormenor.

Descrevem-se de seguida algumas das variáveis auxiliares utilizadas na implementação da funcionalidade deste método:

- Vector dadosTabela:

Vector com um número de elementos igual ao número de exames do tipo Image existentes. Cada elemento é um vector linhaTabela.

- Vector linhaTabela:

Vector com 4 elementos do tipo String, em que cada um deles corresponde ao valor do atributo PatientID, PatientsName, PatientsBirthDate, Modality, respectivamente.

- Vector atributosExames:

Vector com um número de elementos igual ao número de exames do tipo Image existentes. Cada elemento é um objecto da classe Attributes.

- Vector filesExames:

Vector com um número de elementos igual ao número de exames do tipo Image existentes. Cada elemento é do tipo File.

- Vector frameTime:

Vector com um número de elementos igual ao número de exames do tipo Image existentes. Cada elemento é uma String que contém o valor do atributo frameTime de cada um dos exames.

- Vector numFrames

Vector com número de elementos igual ao número de exames do tipo Image existentes. Cada elemento é uma String que contém o valor do atributo NumberOfFrames de cada um dos exames

O código criado segue um raciocínio descrito pelo fluxograma seguinte:

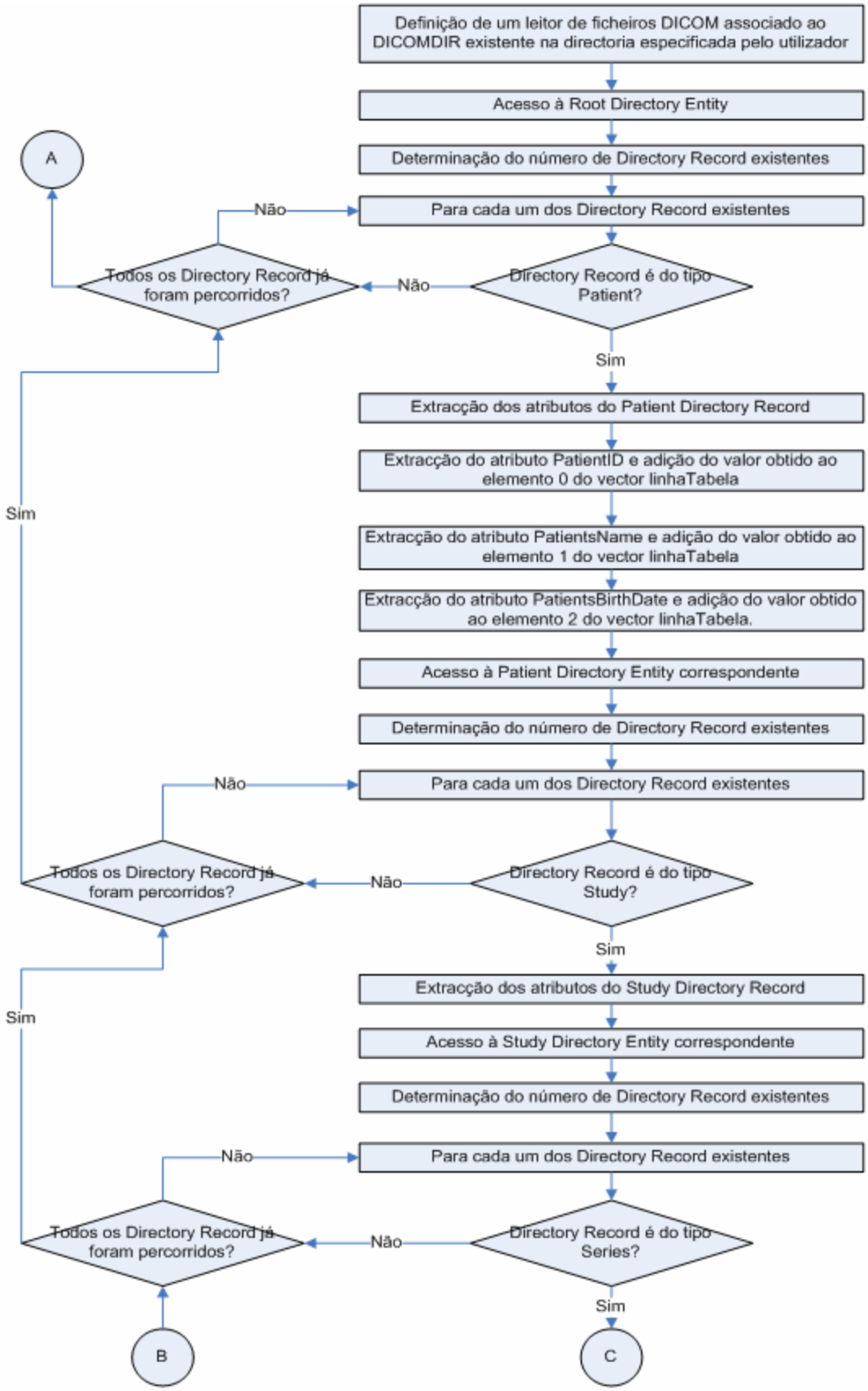

Figura1: Fluxograma do código implementado para o método lê directorio da classe ReadDicomDir.

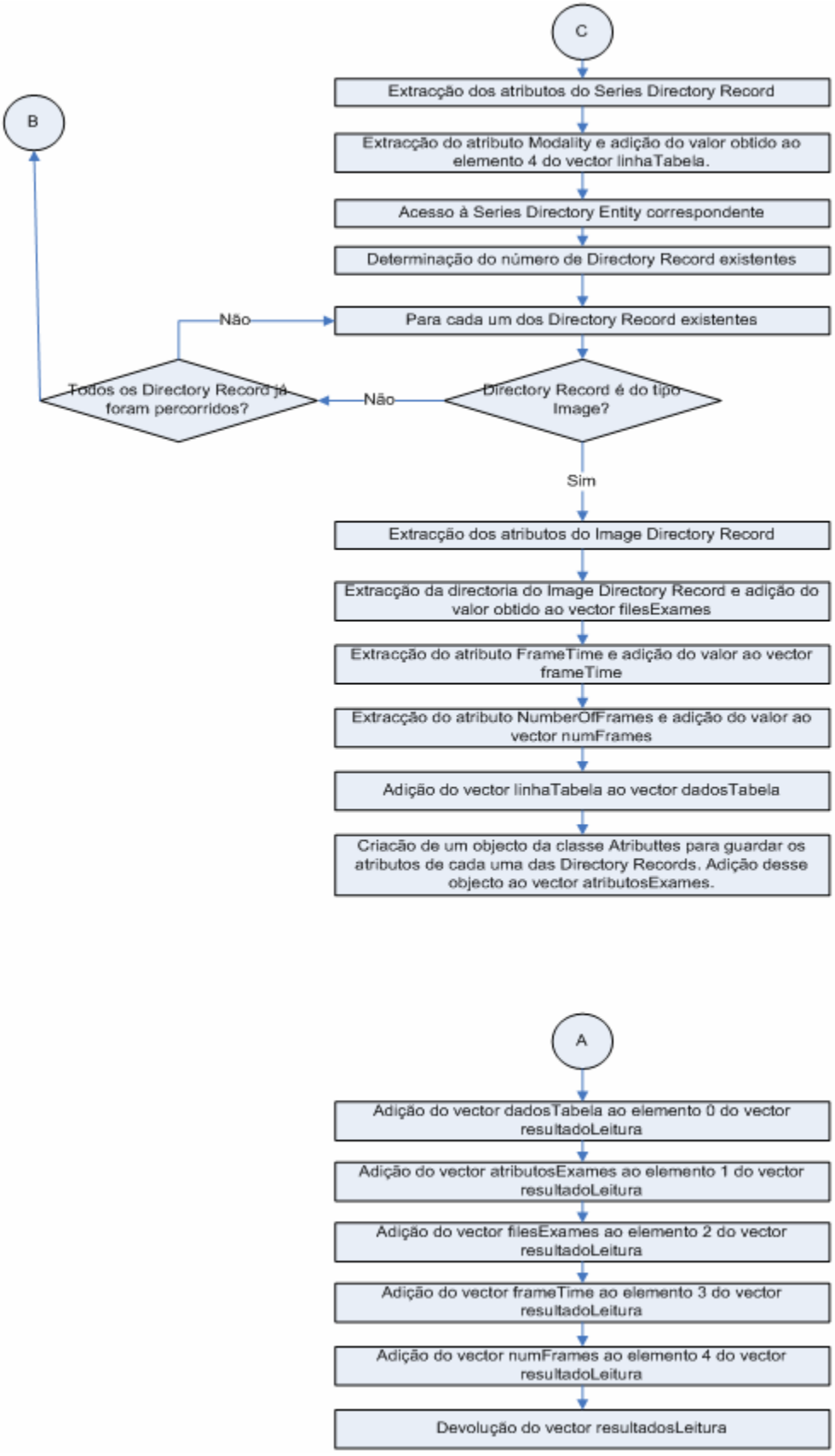

Figura 2: Continuação do fluxograma do código implementado para o método lê directorio da classe ReadDicomDir.

- **class visualizadorAnimacao**

A classe visualisadorAnimacao foi construída tendo por base o código do exemplo *TestRead2.java* da documentação da *dicom-imageio-tools.*

Esta classe implementa uma nova *thread*, na qual é executado o código correspondente à exibição do conjunto de imagens que constituem um exame. Optou-se por exibir esta sequência de imagens numa janela independente da janela principal da aplicação de forma a que o tamanho potencialmente diferente das imagens que se poderiam visualizar não influenciasse o aspecto do interface gráfico da janela principal da aplicação, o que poderia ocorrer caso a visualização fosse embutida na janela principal.

As imagens correspondentes ao Image Directory Record do exame seleccionado na tabela tableExames são extraídas e forma-se um array contendo todas as imagens. Posteriormente esse array é percorrido elemento a elemento e essas imagens são visualizadas sequencialmente, intervaladas por um período de tempo igual ao valor do atributo FrameTime para esse exame.

É exibido no título da janela o valor do atributo PatientId, PatientsName e Modality referente às imagens do exame exibido.

IV. Resultados

Nos ficheiros resultado1.wmv e resultado2.wmv enviados em anexos são mostrados dois exemplo da utilização da aplicação.

Em resultados1.wmv mostra-se a utilização da aplicação desenvolvida quando um utilizador indica a directoria onde se encontra o DICOMDIR, procede à listagem dos exames do tipo imagem existentes, selecciona consecutivamente vários deles de forma a consultar os seus atributos e a exibir a sequência de imagens que os constituem.

Em resultado2.wmv mostra-se o que ocorre quando o utilizador pretende visualizar imagens de um exame sem proceder previamente à selecção de um exame na tabela. Na parte final do vídeo verifica-se a ocorrência de um erro no programa que não conseguimos corrigir. Trata-se de um erro que surge sempre que pretende-se visualizar num curto período de tempo exames com um número elevado de frames (imagens).

V. Conclusão

Analisando os resultados que se obtiveram, conclui-se que os objectivos do trabalho foram alcançados com sucesso. Apenas não se conseguiu resolver o erro que por vezes ocorria no programa e que se descreveu anteriormente.