

Universidade do Minho
Licenciatura em Engenharia Informática
Laboratórios de Informática III



Projecto 2

Ano Lectivo de 2009/2010

54738 **João Gomes**
54745 **André Pimenta**
54825 **Daniel Santos**

13 de Junho de 2010

Resumo

Este relatorio tem como principal objectivo justificar e apresentar os motivos pelas escolhas feitas nas estruturas de dados escolhidas para o projecto.

Será tambem apresentada a diferença de StreamObject para Stream de texto no uso da base de dados, e qual a nossa escolha para o projecto.

Por fim será apresentado um manual do projecto SoAmigos.

Conteúdo

Conteúdo	1
0.1 Motivação e objectivos	1
0.2 Estrutura	1
1 Desenvolvimento	2
1.1 Estrutura de Dados	2
1.1.1 Escolha da estrutura de dados	2
1.1.2 Funcionamento Geral	2
1.2 ObjectOutputStream vs ObjectDeTexto	3
1.3 Manual de utilização	6
2 Conclusão	16
3 Elementos do Grupo	17

0.1 Motivação e objectivos

Os principais motivos e objectivos desta etapa foi basearam se na utilização de uma interface gráfica em JAVA usando swing. Pois com esta ferramenta poderemos construir uma interface bonita e fácil de usar para o utilizador o que se revela muito útil no desenvolvimento de projectos.

Sendo esta etapa uma continuação da etapa anterior pode ser vista como uma continuação do "conhecer" melhor a linguagem JAVA o que se tornou motivante para o grupo.

0.2 Estrutura

O presente relatório é constituído por três capítulos: **Introdução**; **Desenvolvimento**, dividido em três secções, **Estrutura de Dados**, **ObjectStream vs ObjectDeTextoe** **Manual de Utilização** e, por fim, a **Conclusão**.

Capítulo 1

Desenvolvimento

1.1 Estrutura de Dados

1.1.1 Escolha da estrutura de dados

A escolha do tipo de colecção em que guardamos os nossos perfis e as suas mensagens recaiu sobre a estrutura TreeMap.

A necessidade de ter os dados permanentemente ordenados e ter um acesso rápido a esses dados foram sem dúvida as condições que mais influenciaram a nossa escolha.

O tipo de colecção TreeMap é um tipo de colecção que consegue ter estes dois aspectos, dados sempre ordenados e ter um acesso relativamente rápido.

Uma solução alternativa seria usar a colecção HashMap. Este tipo de solução seria muito boa em relação a tempo de acesso, melhor ainda que a solução em TreeMap, visto que o tempo de acesso em TreeMap é no pior caso $O(N)$, o que não acontece com HashMap em que o tempo de acesso é sempre (1). No entanto a solução HashMap deixa a desejar em termos de ter os dados ordenados. Quando queremos obter os dados de forma ordenada temos de copia-los para uma outra colecção e inserir-los de forma ordenada ou então ordena-los depois de os inserir. Como podemos prever esta torna-se uma solução incomportável se virmos que o nosso programa vai estar constantemente a devolver listas de dados que queremos que estejam ordenados. Logo esta opção não servia.

Assim e depois destas considerações decidimos que a opção TreeMap seria a melhor para este caso específico pois consegue ser uma solução que combina rapidez no acesso e na inserção mas o seu melhor atributo é sem dúvida a possibilidade de ter os dados sempre ordenados.

No só amigos optamos por ter os códigos nif, que são os códigos usados nas colecções, como String e assim conseguimos aproveitar mais uma das funcionalidades já fornecidas pela linguagem que são os “comparators” que já são fornecidos para classes primitivas como são exemplo as classes String ou Integer.

Muitas destas conclusões foram observadas através da milestone anterior em que tivemos a oportunidade de ver quais eram as colecções que se comportavam melhor em diferentes situações.

1.1.2 Funcionamento Geral

Como já foi dito o nosso program usa TreeMap para guardar os dados dos utilizadores e as suas mensagens. Decidimos então criar duas TreeMap uma para o nome e a outra para o

nif, partilhando as duas a mesma informação(se uso de clone) e desta forma conseguimos ter acesso organizado aos dados tanto pelo nome como pelo nif.

Estas TreeMaps irão guardar perfies dos utilizadores, e cada perfil contem os dados pessoais de cada utilizador, um conjunto de amigos e as mensagens recebidas.

Os amigos estão guaradados num TreeSet, pois não existem amigos repetidos e apresentação de dados será por ordem de "antiguidade" da ligação entre os utilizadores, os amigos são guardados pelo nif, no entanto desenvolvemos metodos que permite aceder criar e eliminar ligações tanto através do nome como do nif.

As mensagens revelam uma implementação mais complexa. Cada utilizador terá um TreeMap com as mensagens correspondentes a cada utilizador que lhe envio mensagens, ou seja no TreeMap será guardado um conjunto até viente mensagnes(numero maximo de mensagnes por nós definido que é guardado na caixa de correio) de cada utilizador na sua TreeMap que guarda as mensagens recebidas, sendo que a chave nesta TreeMap é o nif do remetente.

Para guarda os dados no fim de cada utilização optamos pelo uso de ObjectOutputStream, que nos permitirá guardar toda a informação de uma só vez, ou guardará tanto os peries, como as suas mensagens como os seus amigos, a escolha de este tipo para guradar os dados será exxplicado mais á frente.

1.2 ObjectOutputStream vs ObjectDeTexto

Depois de realizaras as experiências relativamente à leitura/escrita das bases de dados do programa utilizando streams de objecto e streams de texto não deixam margens para duvidas de qual é o método mais eficiente. Tanto na leitura como a escrita as streams de texto são claramente mais rápidas do que as streams de objectos, como podemos ver através dos dados apresentados. Estas diferenças poderiam ser no entanto um pouco diferentes pois se tivéssemos uma estrutura mais complicada certamente o parser que teríamos de fazer para o ficheiro de texto levaria a que o tempo de leitura aumenta-se um pouco. A Stream de texto revelou se mais rápida pois apenas foi testada a guardar os dados pessoais de cada utilizador, não sendo tendo assim influencia as mensagens e os amigos no seu tempo, mas que com o aumento destes ao longo do uso do programa os tempos aproximariam se mais. Num projecto em que as estruturas fossem mais complexas poderia-mos perfeitamente utilizar streams de objectos visto que o trabalho necessário para desenvolver mecanismos de leitura e escrita seriam bastante simples pois a leitura e escrita são directas o que não aconteceria com as streams de texto em que temos de fazer o parser pois não existem métodos que se adaptem a todas as estruturas. Apesar de todo optamos por escolher objectStream pois facilitou a programação, pois teríamos de guardar para além dos utilizadores também as mensagens e os amigos em texto, e com objectStream é guardado tudo automaticamente, e porque estes tempos foram considerados para base de dados com apenas informações de perfis, sendo que o resto dos dados não foram tidos em contam e que se certa forma alterariam os desempenhos apresentados.

Estas conclusões foram retiradas atraves dos seguintes dados:

		Numero de utilizadores carregados				tempo em (ms)
		5000	10000	15000	18000	
StreamObject	StreamObject	358	719	1069	1189	
	StreamsTexto	16	36,34	59,03	79	

Figura 1.1: Tabela de dados Carregar ficheiro

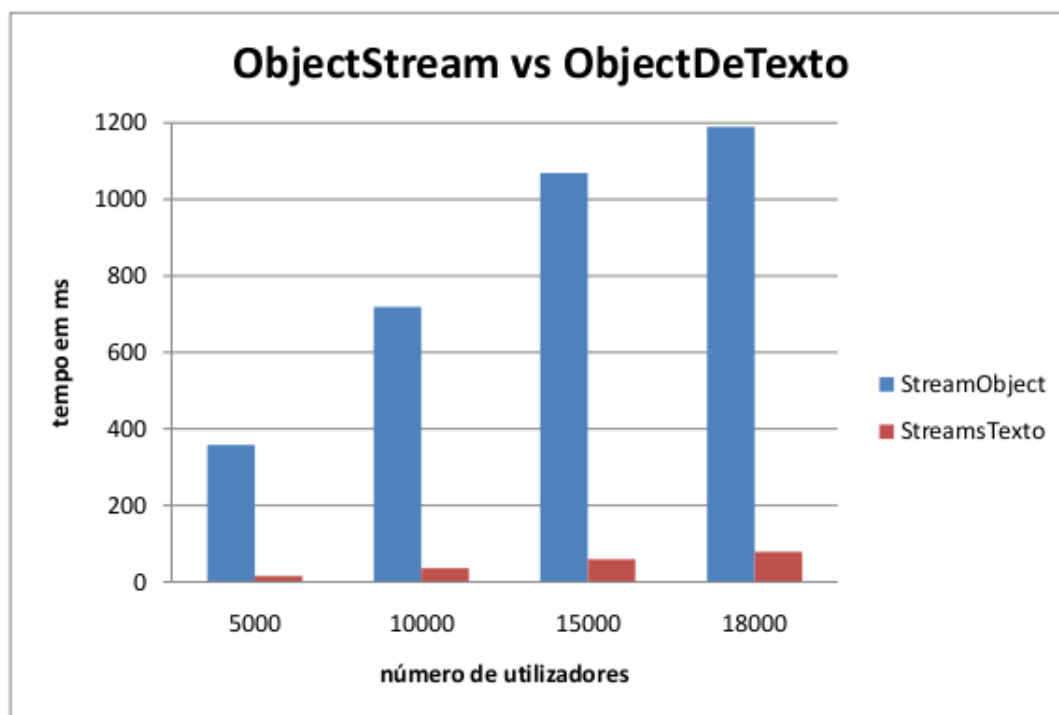


Figura 1.2: Grafico de dados Carregar ficheiro

Numero de utilizadores carregados					tempo em (ms)
	5000	10000	15000	18000	
StreamOb	700	1318	2102	2298	
Streamste	11	21	29	36	

Figura 1.3: Tabela de dados Gravar ficheiro

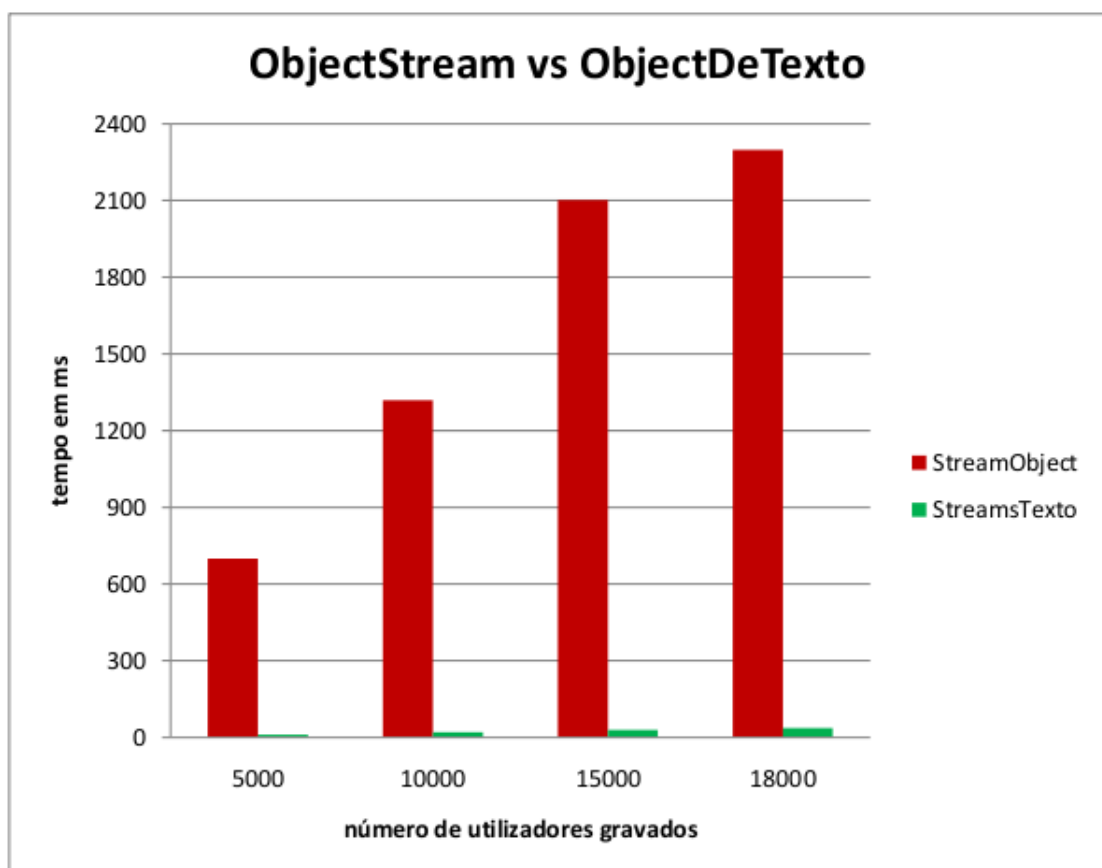


Figura 1.4: Grafico de dados Gravar ficheiro

1.3 Manual de utilização

O programa apresenta de uma forma muito simples e fácil de usar , sendo que todas as suas funcionalidades se tornam evidentes à primeira vista até para utilizadores menos experientes. Ao iniciar o SoAmigos será apresentado um Menu que permitirá iniciar sessão, criar uma nova conta caso ainda não esteja registado, ter acesso as funcionalidades de administrador e fechar o programa.



Figura 1.5: Menu Inicial

Como estamos a usar pela primeira vez o programa vamos nos registar, e para tal selecionamos o segundo botão que diz Registrar.



The image shows a window titled "Adicionar Utilizador" with a standard Windows-style title bar (minimize, maximize, close buttons). The window has a light gray background. In the center, there is a registration form. It consists of three labels on the left: "Nome:", "Morada:", and "NIF:". To the right of each label is a white rectangular input field. To the right of the input fields are two blue buttons with white text. The top button is labeled "Confirmar" and the bottom button is labeled "Cancelar".

Figura 1.6: Registrar

Introduzimos o nosso nif, nome e morada e depois clicamos no botão confirmar, e já temos a conta criada.

Após termos conta criada iremos iniciar sessão, carrega mos então no prim eiro botão que diz "iniciar sessão" e será apresentada uma caixa a pedir o nome e nif(terão de cõnsidir com os nossos dados, para poderemos ter acesso a nossa conta), inserimos os dados e confirmamos e iniciaremos a nossa conta.

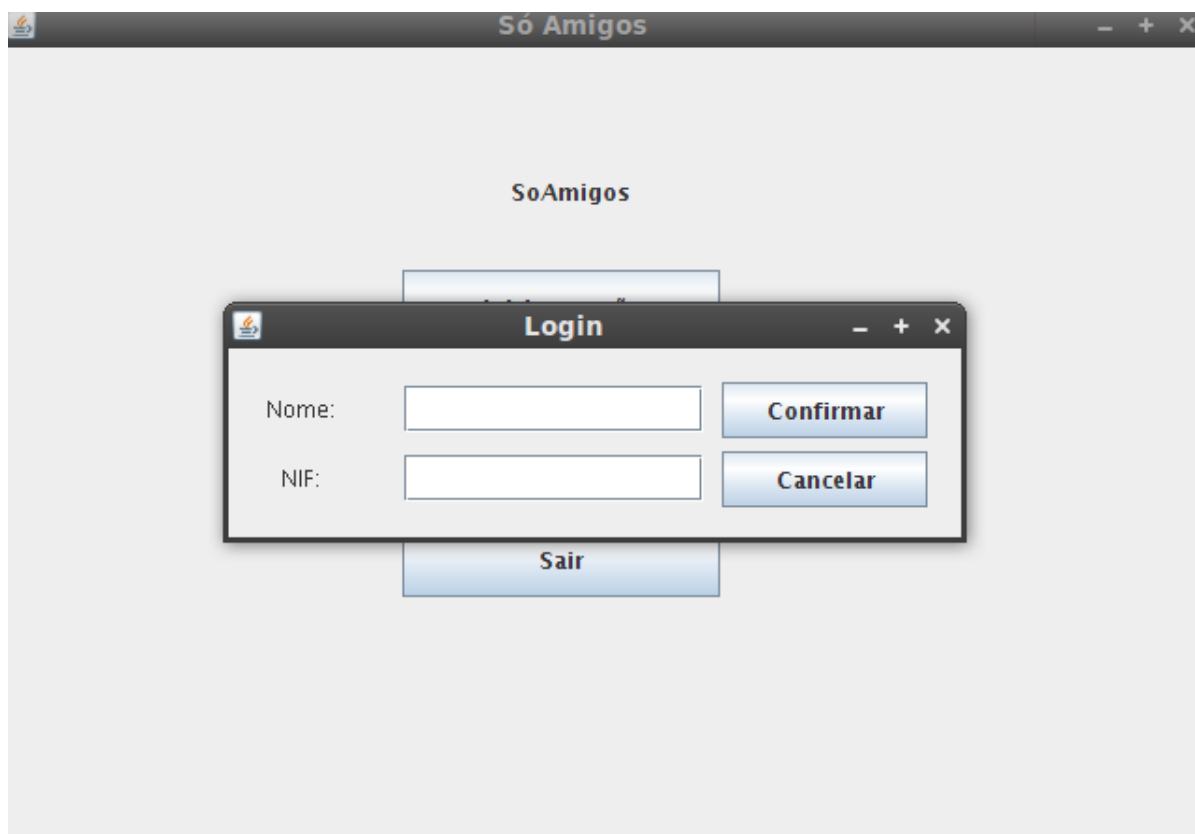


Figura 1.7: Login

Agora dentro da nossa conta poderemos ter acesso a mensagens aos nossos dados pessoais, aos nossos amigos e poderemos pesquisar utilizadores.

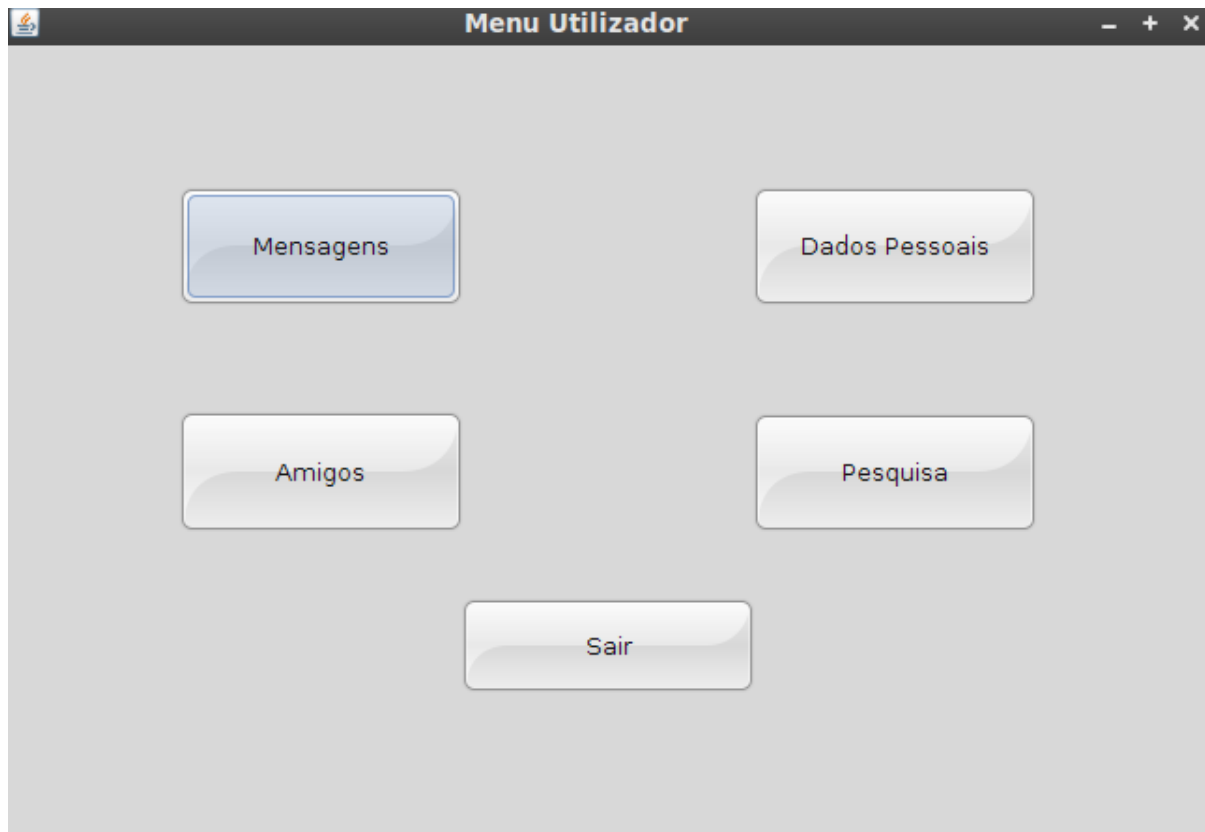


Figura 1.8: Menu do utilizador

Iremos então usar botão mensagens, este vai nos dar acesso à nossa caixa de entrada de mensagens, onde podemos escolher o remetente das mensagens, e depois ver todas as mensagens desse remetente, navegando nos botões anterior e seguinte e apaga las no botão apagar.

O botão enviar dá acesso a Menu que permite enviar mensagens, onde escolhemos para quem queremos enviar e aí temos duas opções ou por nome ou por nif, um espaço para o assunto e uma caixa para escrever o texto que se pretende enviar

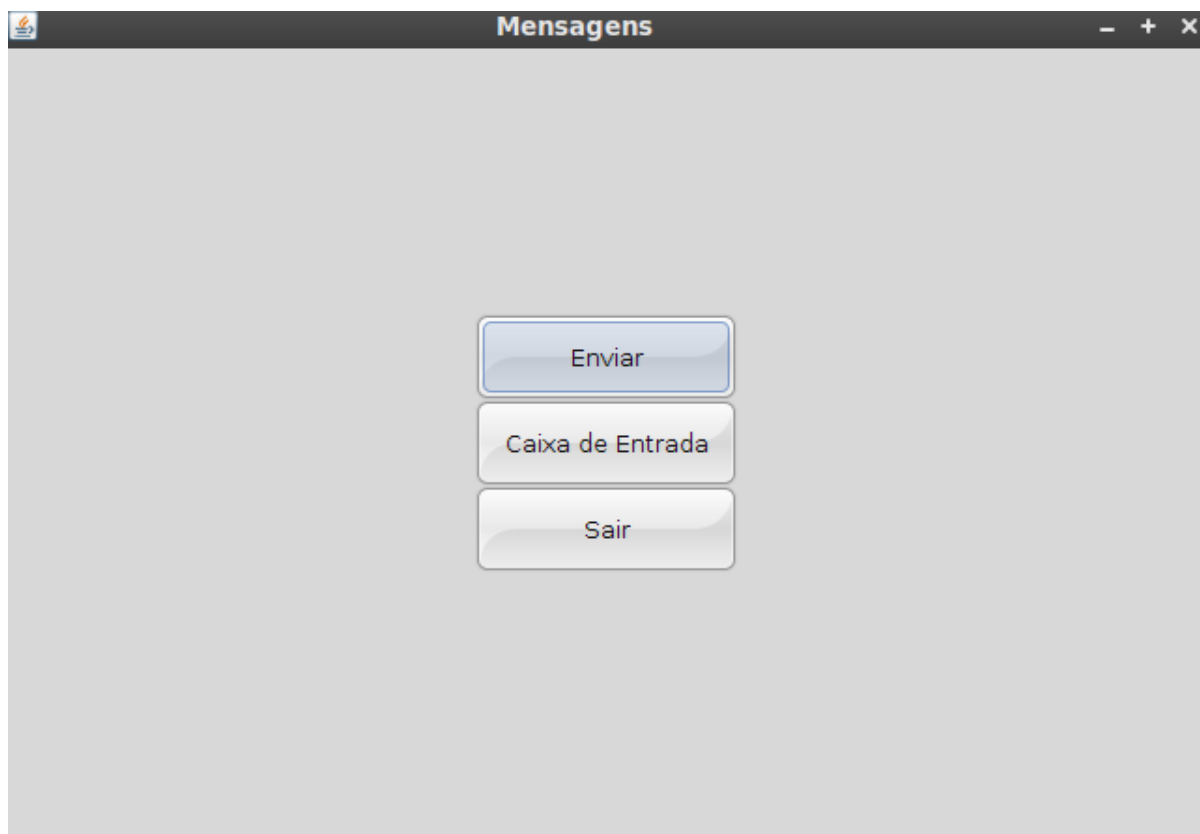


Figura 1.9: Menu de mensagens

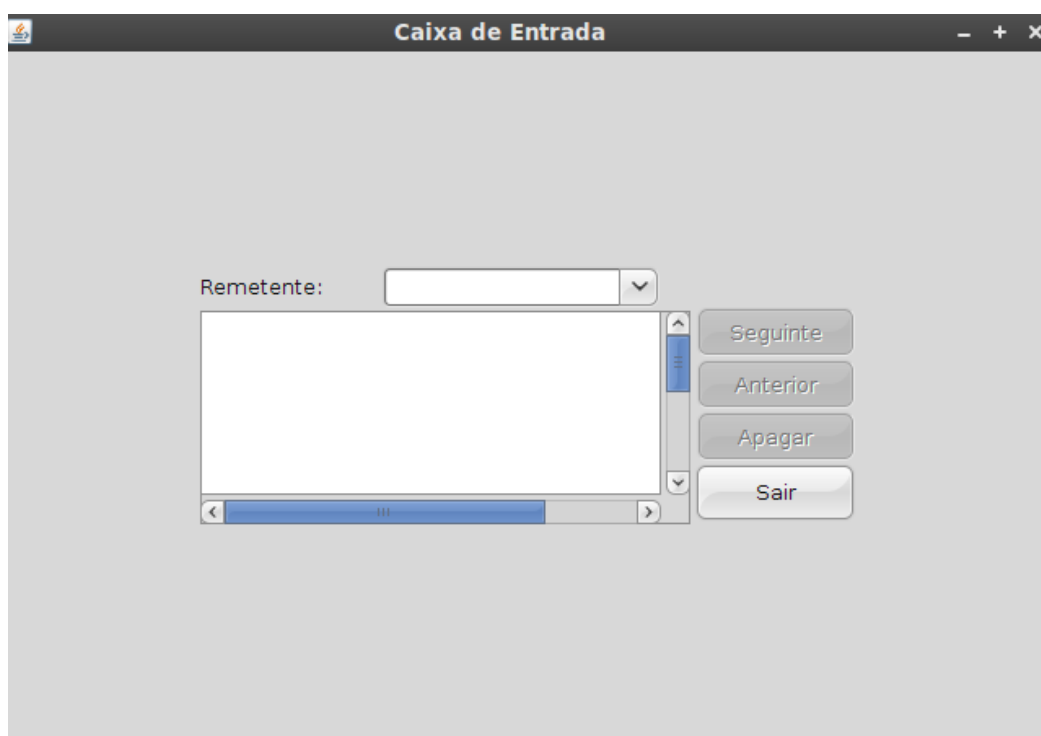


Figura 1.10: Caixa de Entrada



Figura 1.11: Enviar mensagem

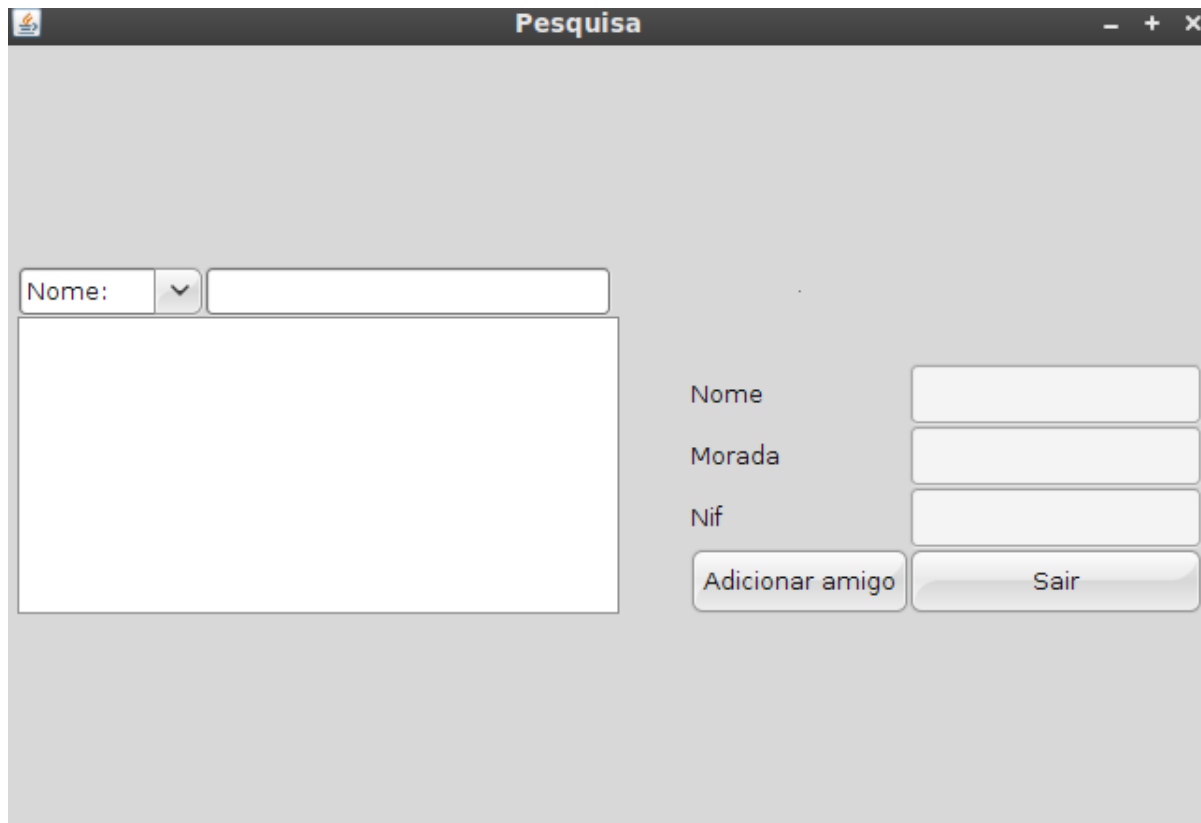
Voltando ao Menu principal se escolhermos Amigos, temos acesso ao Menu de amigos, onde podemos ver todos os nossos amigos que serão apresentados por ordem de ligação de amizade, e podemos também adicionar amigos tanto por nif ou nome remover de amigo, ou enviar mensagem para este.



The image shows a graphical user interface window titled "Amigos". The window has a standard title bar with a small icon on the left and window control buttons (minimize, maximize, close) on the right. The main area of the window is light gray. On the left side, there are four input fields stacked vertically. The first field is preceded by a dropdown menu labeled "Nome". The second field is preceded by the label "Nome:". The third field is preceded by the label "Morada". The fourth field is preceded by the label "NIF:". To the right of these input fields, there is a vertical stack of four buttons: "Adicionar", "Remover", "Enviar mensagem", and "Sair". Below the input fields, there is a large, empty rectangular box.

Figura 1.12: Menu Amigos

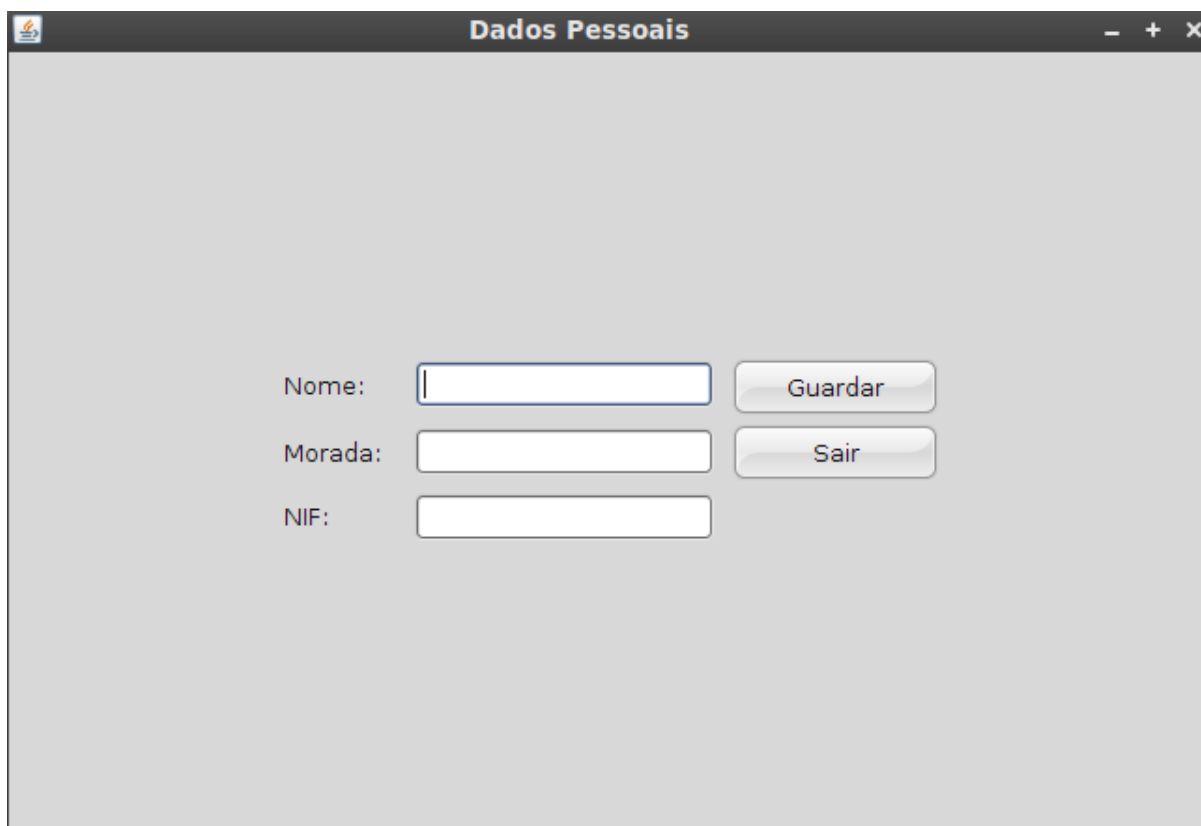
Voltando ao Menu principal e agora usando as pesquisas. Neste Menu termos acesso a todos os utilizadores e podem ser feitas pesquisas através do nome ou do nif, e poderemos adicionar pessoas como amigos.



The image shows a window titled "Pesquisa" with a standard macOS-style title bar (minimize, maximize, close buttons). The interface is divided into two main sections. On the left, there is a search form with a label "Nome:" followed by a dropdown arrow and a text input field. Below this is a large, empty rectangular box, likely for displaying search results. On the right, there are three vertically stacked text input fields labeled "Nome", "Morada", and "Nif". At the bottom right, there are two buttons: "Adicionar amigo" and "Sair".

Figura 1.13: Menu pesquisa

Falta agora usar o botão Dados pessoais que nos permite ver e editar os nossos dados pessoais, e caso se decida editar basta apenas clicar no botão guardar e a informação será guardada.



The image shows a web application window titled "Dados Pessoais". Inside the window, there are three input fields for personal data: "Nome:", "Morada:", and "NIF:". To the right of the "Nome:" field is a button labeled "Guardar". To the right of the "Morada:" field is a button labeled "Sair". The "NIF:" field does not have a button next to it. The window has a standard OS title bar with minimize, maximize, and close buttons.

Figura 1.14: Menu Dados pessoais

Por fim falta usar o botão Admin acessível no menu inicial. Neste Menu teremos acesso a todos os utilizadores do programa e poderemos pesquisar utilizadores dentro da base de dados e apagar contas de utilizadores dentro do programa.

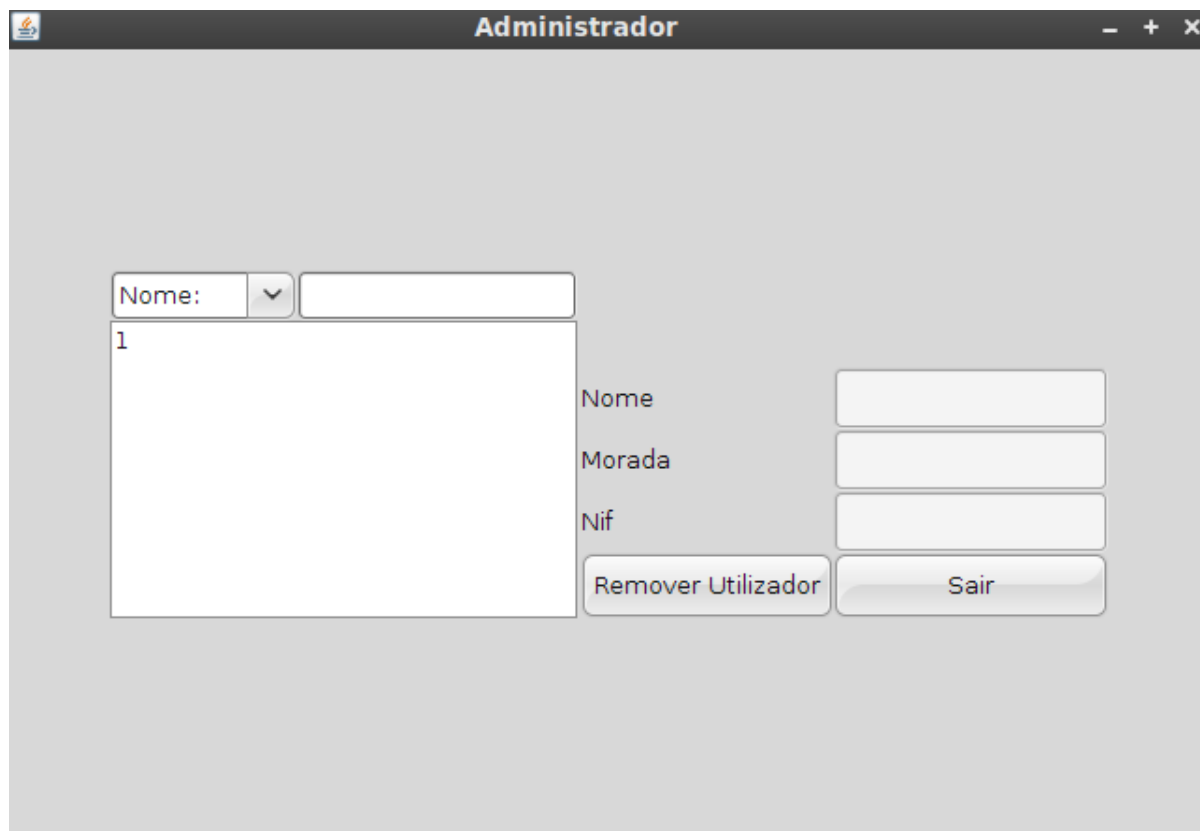


Figura 1.15: Menu Administrador

Capítulo 2

Conclusão

Após a conclusão final do projecto 2 ("soAmigos") de li3 podemos afirmar que a construção gráfica da interface do programa se revelou fácil e muito útil. Podendo até em forma de comparação com a utilizada no projecto1 afirmar que é mais bonita e agradável ao utilizador, para além de ser mais fácil de desenvolver.

Conseguimos estruturar o programa de forma eficaz, tendo este um bom desempenho e oferece ao utilizador um leque de opcional capaz de cobrir todas as suas necessidades.

O que podemos apontar como o ponto mais fraco do projecto por nós desenvolvido é a forma de gravar os dados. Optamos pelo uso de ObjectOutputStream que se revela bastante mais lento do que em texto, apesar de não ser muito notável ao utilizador com bases de dados enormes pode causar algum desconforto ao iniciar e fechar o programa.

A nível pessoal podemos afirmar que a elaboração deste projecto foi muito útil aos elementos do grupo pois ajudou a compreender melhor o uso da linguagem JAVA, e a conhecer as diferenças de utilização entre PI e POO.

Finalizando, o projecto apresenta-se com um bom nível de usabilidade e desempenho, cumprindo todas as necessidades pedidas pelo utilizador.

Capítulo 3

Elementos do Grupo



Figura 3.1: João Miguel



Figura 3.2: André Pimenta

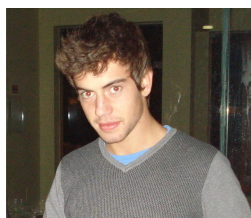


Figura 3.3: Daniel Santos