UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE DARCY RIBEIRO
LABORATÓRIO DE ENGENHARIA E EXPLORAÇÃO DE PETRÓLEO

BANCO DE DADOS DE FLUIDOS DE PERFURAÇÃO

TRABALHO DA DISCIPLINA PROGRAMAÇÃO PRÁTICA

ANNA MARA CORRÊA DE OLIVEIRA
RAIZA GOMES DE SOUZA

MACAÉ - RJ
Março - 2016

Sumário

Capítulo 1

Introdução

No presente trabalho desenvolve-se um projeto de engenharia em linguagem orientada a objeto que tem como principal objetivo o gerenciamento de informações de fluidos de perfuração desenvolvidos no Laboratório de Fluidos do LENEP-Laboratório de Engenharia e Exploração de Petróleo a partir da construção de um banco de dados. Dessa forma a principal finalidade do programa é desenvolver um banco de dados onde será armazenado infromações relacionadas às propriedades físicas e químicas dos fluidos, onde o usuárrio terá acesso a essas informações assim como poderá incluir novas informações a respeito de novos fluidos desencolvidos.

Este programa tomou como base o código do aluno João Ricardo Côre Dutra. Seu trabalho encontra-se em desenvolvimento e é um Banco de Dados de Geoquímica (BDGQ).

1.1 Escopo do Problema

Banco de dados são gerenciadores de grandes grupos de informações. Esse gerenciamento consiste em definir a estrutura para o armazenamento de informações e o fornecimento de mecanismos para manipulá-las. Esse gerenciamento é executado pelos Sistenas de Gerenciamento de Banco de Dados (SGBD), softwares que possuem recursos para efetuar esse tipo de manipulação.

Os fluidos de perfuração são misturas complexas. Sua reprodução requer o armazenamento de dados como tipo de base, teor de base, propriedades físicas e químicas, propriedades reológicas e aditivos no computador. A melhor maneira de gerenciar estas informações é através do uso de um banco de dados.

1.2 Objetivos

Os objetivos deste trabalho são:

- Objetivo geral:

  - Desenvolver um banco de dados contendo informações a respeito de fluidos de perfuração desenvolvidos no Laboratório de Fluidos do LENEP - Laboratório de Engenharia e Exploração de Petróleo.

- Objetivos específicos:

  - Possibilitar inclusão de informações de novos fluidos de perfuração;
  - Possibilitar acesso às informações do banco de dados pelo usuário;
  - Possibilitar realização de pesquisa de informações no banco de dados pelo usuário.

Capítulo 2

Especificação

Apresenta-se neste capítulo a especificação do sistema a ser modelado e desenvolvido.

2.1 Especificação do Programa

Deseja-se desenvolver neste projeto de engenharia um programa com interface em modo texto que permita ao usuário ter acesso a um banco de dados de fluidos de perfuração com diferentes propriedades físicas e químicas. O usuário poderá incluir e excluir novos fluidos de perfuração ao banco de dados, listar e exibir fluidos de perfuração, realizar pesquisa de fluidos com propriedades especificas determinadas pelo usuário e gerar relatórios. Também poderá excluir fluidos.

Para isso o usuário terá acesso a um menu com opções : banco de dados, abrir diretório de trabalho, abrir diretório de dados e configurações. Ao selecionar a opção Banco de Dados, o usuário terá acesso ao menu com as opções de listar, pesquisar, inserir, excluir, exportar e informações de um determinado fluido de perfuração.

Por ser um software científico o mesmo deve ser passível de modificações, logo dever ter o seu código aberto (licença 6PL2). O banco de dados será desenvolvido em linguagem de programação orientada a objeto C++ e será utilizado em laboratórios do LENEP / CCT/ UENF - Macaé - RJ para fins estudantis, laboratoriais e de pesquisa.

2.2 Casos de Uso do Programa

Um caso de uso descreve um ou mais cenários de uso do software, exemplos de uso, como o sistema interage com usuarios externos (atores). Ademais, ele deve representar uma sequência típica de uso do programa (a execução de determinadas tarefas-padrão). Tambem deve representar as excecões, casos em que o usuário comete algum erro, em que o sistema não consegue realizar as tarefas solicitadas.

Apresenta-se a seguir alguns diagramas de caso de uso. O objetivo é gerar uma percepção básica das interações do usuário com o banco de dados.

2.3 Diagrama de Caso de Uso Geral

Abaixo encontra-se o diagrama de caso de uso geral que descreve a interação do usuário com o banco de dados.

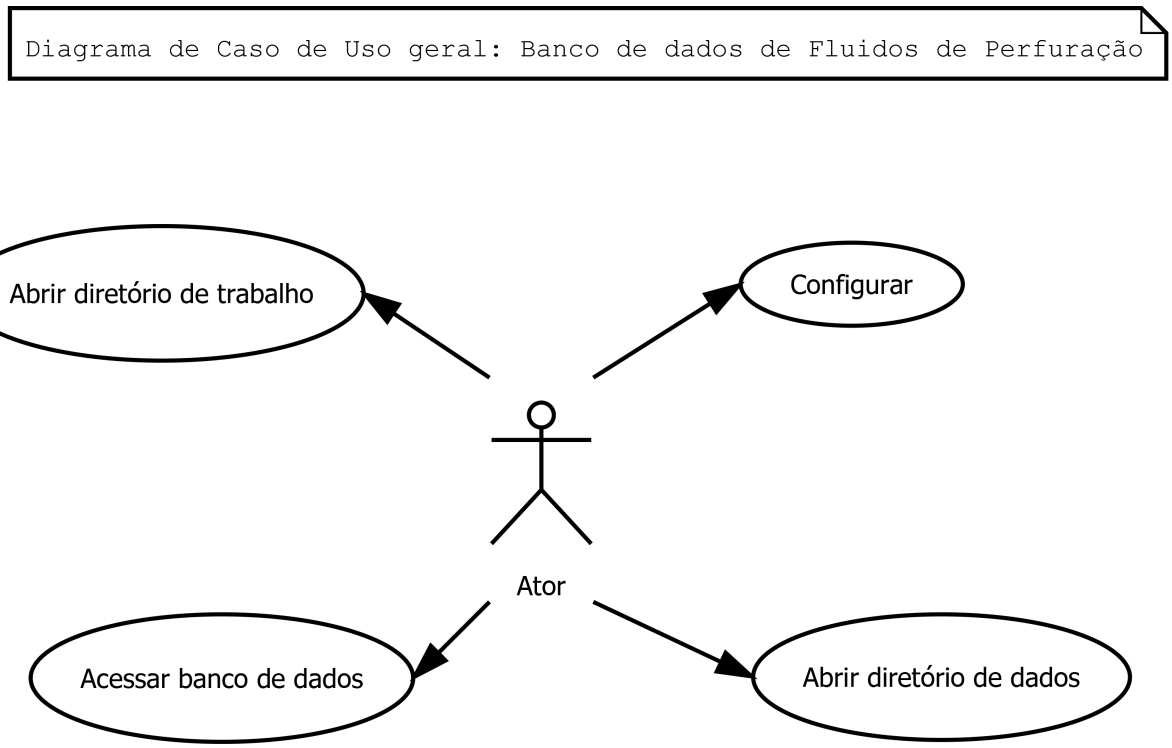


Figura 2.1: Diagrama de caso de uso geral

2.4 Diagramas de Caso de Uso Específicos

Na figura 2.2 encontra-se o diagrama de caso de uso especifico “Acessar Banco de Dados”

Na figura 2.3 encontra-se o diagrama de caso de uso especifico “Inserir Fluido de Perfuração”

Na figura 2.4 encontra-se o diagrama de caso de uso especifico “Pesquisar Fuidos com propriedades específicas”

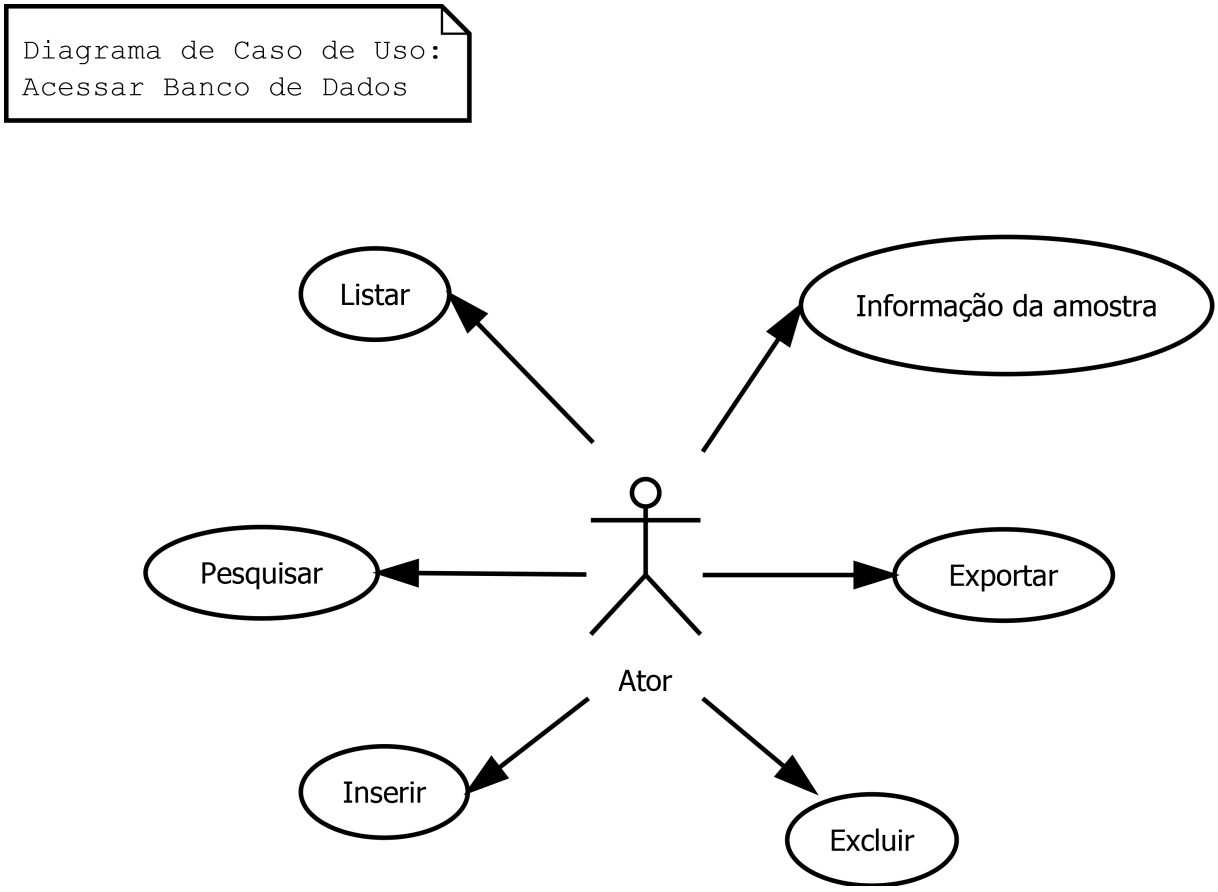


Figura 2.2: Diagrama de Caso de uso específico: Acessar banco de dados

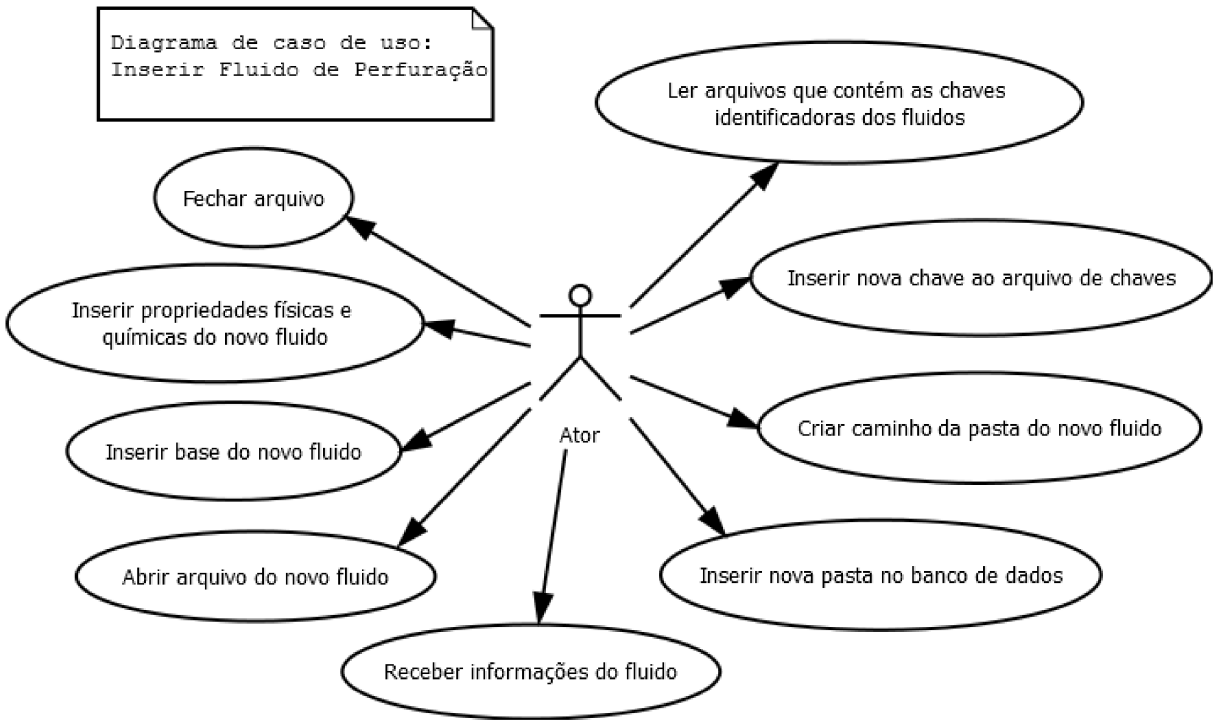


Figura 2.3: Diagrama de caso de uso específico: Inserir Fluido de Perfuração

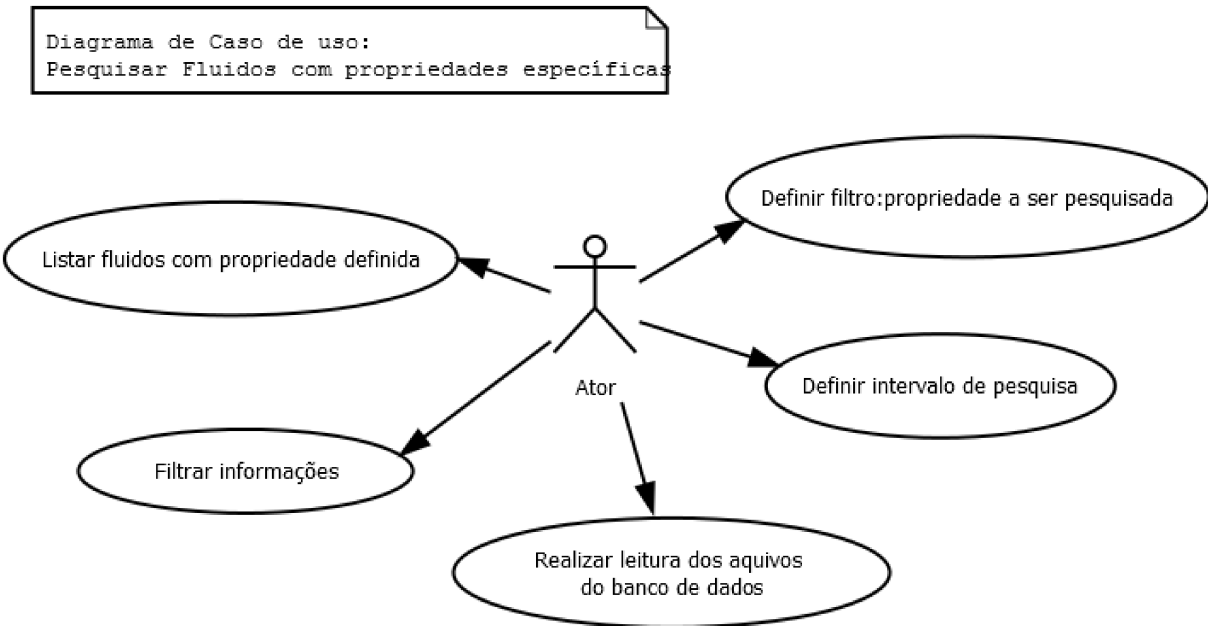


Figura 2.4: Diagrama de caso de uso específico: Pesquisar Fuidos com propriedades específicas

Capítulo 3

Elaboração

No processo de elaboração, é realizado um estudo de abrangência do software em desenvolvimento, ajustando os requisitos iniciais do sistema que foram definidos na etapa de especificação. Tem-se como objetivo possibilitar o desenvolvimento de um sistema útil, que atenda as necessidades do usuário e permita futuras extensões do programa.

3.1 Análise de Domínio

A análise de domínio é uma parte da elaboração; seu objetivo é entender o domínio, a abrangência do sistema a ser desenvolvido. Envolve itens como estimar o reuso do software utilizando-se da criação de bibliotecas genéricas. Neste ponto, o analista pensa no sistema de uma forma mais genérica, identificando conceitos fundamentais que podem ser reaproveitados em outros sistemas.

Considerando que o software tem como objetivo principal a criação de um banco de dados de fluidos de perfuração desenvolvidos no Laboratório de Engenharia e Exploração de Petróleo-LENEP, base óleo ou base água, que contém as características de acordo com suas composições quimicas e tipos de aditivos, a análise de domínio envolverá:

- O uso dos livros de engenharia de reservatório, de programação orientada a objeto, de fluidos de perfuração para desenvolvimento do software.
- Consultas a livros, sites, artigos, monografias e dados de trabalhos de iniciação científica dos alunos do LENEP.
- Como o programa tem interface bastante simplificada e realiza rotinas que dependem dos dados de entrada e dados armazenados, a necessidade de disco rígido é proporcional ao tamanho da base de dados.

3.1.1 Banco de Dados

Um SGBD tem que ter algumas particularidades e deve facilitar o processo de definir (especificar tipos de dados a serem armazenados), construir (armazenar dados que possam

ser manipulados por um SGBD) e manipular (inserir, atualizar e remover base dados de diversas aplicações). As principais características de um SGBD são:

- Controle de redundância: pode-se construir regras para que o gerenciamento seja mais eficaz, evitando assim a redundância dos dados e economizando espaço em disco. Por exemplo, um aluno só pode ser cadastrado uma única vez em cada curso; cada disciplina só pode ser cadastrada uma vez em um único curso; ou ainda, cada aluno só pode se inscrever uma vez em cada matéria.

- Restrição a acesso não autorizado: Em um banco de dados com vários usuários, cada um tem acesso no que lhe é permitido. Com um SGBD é possível restringir os acessos de cada usuário ou grupo de usuários, permitido assim acessos autorizados para cada usuário.

- Garantia de armazenamento persistente: Com um SGBD é possível armazenar dados de uma forma organizada.

- Garantia de armazenamento de estruturas para o processamento eficiente de consultas: Uma outra característica de um SGBD é que além de armazenar dados ele deve prover mecanismo que facilitem a busca, a inserção ou atualização da base de dados.

- Compartilhamento de dados: SGBDs multiusuários devem fornecer controle de concorrência para assegurar que atualizações simultâneas resultem em modificações corretas.

- Fornecimento de múltiplas interfaces: Devido aos vários tipos de usuários, com variados níveis de conhecimento técnico, um SGBD deve fornecer uma variedade de interfaces para atendê-los. Os tipos de interfaces incluem linguagens de consulta para usuários ocasionais, interfaces de linguagem de programação para programadores de aplicações, formulários e interfaces dirigidas por menus para usuários comuns.

- Representação de relacionamento complexo entre dados: Uma base de dados pode possuir uma variedade de dados que estão inter-relacionados de muitas maneiras. Um SGBD deve ter a capacidade de representar uma variedade de relacionamentos complexos entre dados, bem como recuperar e modificar dados relacionados de maneira fácil e eficiente.

- Backup e restauração: Garantir backup e restauração de dados é tarefa essencial para qualquer SGBD. Mesmo que as falhas sejam ocasionadas por falhas de software ou hardware ele deve garantir a integridade dos dados.

- Restrições de integridade: Num SGBD é possível impor restrições, por exemplo, em uma tabela ALUNO que contém atributos: Nome, CPF, Endereço, Tel, o atributo Nome possa ter no máximo 50 caracteres, e que CPF pode ter 11 caracteres e que

Tel pode receber 11 inteiros, ou ainda, a tabela Turma deve ser preenchida com dados da tabela Professor e da tabela Aluno, etc.

Banco de dados orientado a objeto

Visando acompanhar a tendência da época e também possibilitar resolver as limitações que os bancos de dados possuíam, foi proposto um novo sistema de banco de dados orientados a objeto (BDOO).

De uma forma bem simples pode-se dizer que o BDOO é nada mais que a junção entre conceitos de OO com conceiros de SGBD, ou seja, ele é todo baseado nos paradigmas da OO unido aos objetivos básicos dos SGBD.

3.1.2 Fluidos de Perfuração

Os fluidos de perfuração são vistos de diferentes maneiras por diferentes autores. O instituto Americano de Petróleo (API) considera fluido de perfuração qualquer fluido circulante capaz de tornar a operação de perfuração viável. Autores como Thomas et al. (2001) consideram os fluidos de perfuração como misturas complexas de sólidos, líquidos, produtos químicos e, por vezes, até de gases.

Um fluido de perfuração além de ter de realizar suas funções primordiais, que são a suspensão, o controle de pressão, a estabilização das formações, apresentar poder de flutuação e de resfriamento da broca (Duarte, 2004), também deve apresentar características adequadas para que possam ser utilizados nas diversas formações. Sendo assim, um fluido de perfuração deve ser estável quimicamente, facilitar a separação dos cascalhos na superfície, ser inerte (não reagir) com as rochas produtoras, ser capaz de aceitar tratamento físico e/ou químico, ser passível de bombeamento, deve apresentar baixo grau de corrosão e abrasão (esfoliamento) em relação à coluna de perfuração e a outros equipamentos da coluna de perfuração, e ainda não ser agressivo ao meio ambiente (Thomas et al., 2001). Além das funções cruciais de um fluido de perfuração, eles apresentam funções e características secundárias, tais como: resfriar e limpar pequenas impurezas, apresentar baixo custo de operação, facilitar as interpretações geológicas do material retirado do poço, dentre outras.

Os principais componentes dos fluidos de perfuração são a base (ar, água e óleo) e os aditivos químicos. De acordo com a base predominante utilizada em sua preparação os fluidos são classificados em base ar, água e/ou base óleo. A maior parte das operações de perfuração no mundo usam fluidos (lamas) base água, contra apenas cerca de 5 a 10% que utilizam fluidos base óleo e uma porção ainda menor de poços que são perfurados com fluidos base ar (Caenn, 1995).

Segundo Caenn et al (1995), os aditivos mais comuns utilizados nos fluidos de perfuração são os polímeros, surfactantes, sais e bentonitas. Pesquisas são continuamente executadas para aumentar a performance dos fluidos de perfuração e aditivos são frequente-

mente desenvolvidos para alterar uma ou mais propriedades da lama, para que assim possa ser formulado o fluido que atenda às necessidades exigidas para cada aplicação. Os principais aditivos utilizados são adensantes, sais, redutores de filtrado, biopolímeros, viscosificantes, dispersantes, defloculantes, emulsionantes, biocidas, salmoura, lubrificantes, inibidores de corrosão e controladores de pH (Bleier, 1992; Economides, 1998; Veiga, 1998; Barbosa, 2005; Candler & Friedheim, 2006).

- Adensantes - substâncias usadas para aumentar a densidade com o intuito de controlar a pressão hidrostática do poço para prevenir a ocorrência de blowouts ou o dano à formação. Qualquer substância mais densa que o fluido e que não provoque nenhum efeito adverso nas demais propriedades podem ser usadas. Além do custo, deve-se levar em consideração o volume ocupado pelo aditivo. Os materiais mais usados com essa finalidade são Dolomita, Calcita, Hematita, Galena e especialmente a Barita (BaSO4) (Darley& Gray, 1988).

- Viscosificantes - agentes utilizados para conferir viscosidade alta em baixo cisalhamento e viscosidade baixa em alto cisalhamento. Neste caso, o aditivo mais utilizado é a bentonita, a qual incha em contato com a água reduzindo a fricção entre a coluna de perfuração e as paredes do poço, também podem ser utilizados polímeros sintéticos como o policatiônico;

- Biopolímero- usados no controle reológico e para melhorar o processo de carregamento de cascalhos durante a perfuração, geralmente atuam tornando o fluido mais viscoso. Os polímeros mais utilizados na indústria são: CMC (carboximetilcelulose), HEC (hidroxietilcelulose) e o CMS (carboximetilamido);

- Sais - atuam como inibidores das formações ativas de maneira a reduzir o escoamento hidráulico para a formação, além de estimular o escoamento de água da formação argilosa para o fluido de perfuração. Os sais mais utilizados em fluidos de perfuração base água são: cloreto de sódio (NaCl), cloreto de potássio (KCl) e cloreto de cálcio ($CaCl_2$), entretanto também podem ser utilizados polímeros naturais e sintéticos;

- Salmouras - utilizada como a fase aquosa, tem a função de balancear as interações dos fluidos de perfuração com argilas ou sais solúveis das formações. Normalmente utiliza-se NaCl ou KCl como salmouras para fluidos à base de água e CaCl2 para fluidos sintéticos ou à base de óleo;

- Redutores de filtrado – adicionados com o objetivo de controlar a perda de fluido, atuam minimizando a penetração do fluido de perfuração na formação e promovendo a melhoria do reboco formado nas paredes do poço. Geralmente utiliza-se amido, bentonita, lignita ou polímeros para alcançar tal finalidade;

- Dispersantes - Os aditivos do tipo lignosulfonatos e lignito possuem a função de dispersarem os sólidos presentes nos fluidos de perfuração e, por isso, são conhecidos como dispersantes;
- Defloculantes - com o intuito de prevenir a floculação dos sólidos ativos nos fluidos de perfuração utilizam-se principalmente poliacrilatos de cálcio, sódio e potássio;
- Emulsionantes - tais como os ácidos graxos e alquilados sulfonados, responsáveis por formar, manter e estabilizar emulsões óleo em água e água em óleo;
- Biocidas - aditivos como glutaraldeído, sais quaternários de amônio e tiocianato usados para controlar os processos fermentativos do fluido de perfuração devido à ação de microorganismos;
- Lubrificantes - aplicados para reduzir o atrito entre a coluna de perfuração e as paredes do poço usam-se, por exemplo, ésteres de ácidos graxos e polipropilenoglicol;
- Inibidores de corrosão - entram na formulação do fluido com o intuito de prevenir corrosão e descamação dos tubos e demais equipamentos de perfuração. Para este fim tem-se aminas e álcoois de cadeia longa;
- Controladores de pH – usam-se hidróxidos de sódio ou potássio, ácido acético, acetato e carbonato de sódio como aditivos com função principal de controlar o pH dos fluidos numa faixa preestabelecida, mas também como redutores de corrosão e estabilizadores de emulsões.

3.2 Identificação de Pacotes

Na linguagem de modelagem unificada (UML), um pacote é um mecanismo de agrupamento genérico que contém classes que fazem parte de um assunto e relacionam-se por um conceito comum. Em outras palavras, agrupam classes que se relacionam com maior frequência.

3.3 Diagrama de Pacotes

Os pacotes principais deste software são discutidos a seguir e ilustrado na figura 3.1.

- Pacote Fluido: Contém as propriedades físicas e químicas associadas aos fluidos e as diferentes características associadas ao tipo de base óleo ou água, e aos aditivos.
- Pacote Interface: Contém a estrutura necessária para definição da interface do programa.

- Pacote Banco de Dados: Contém as informações dos diferentes tipos de fluidos e as propriedades relaciondas.

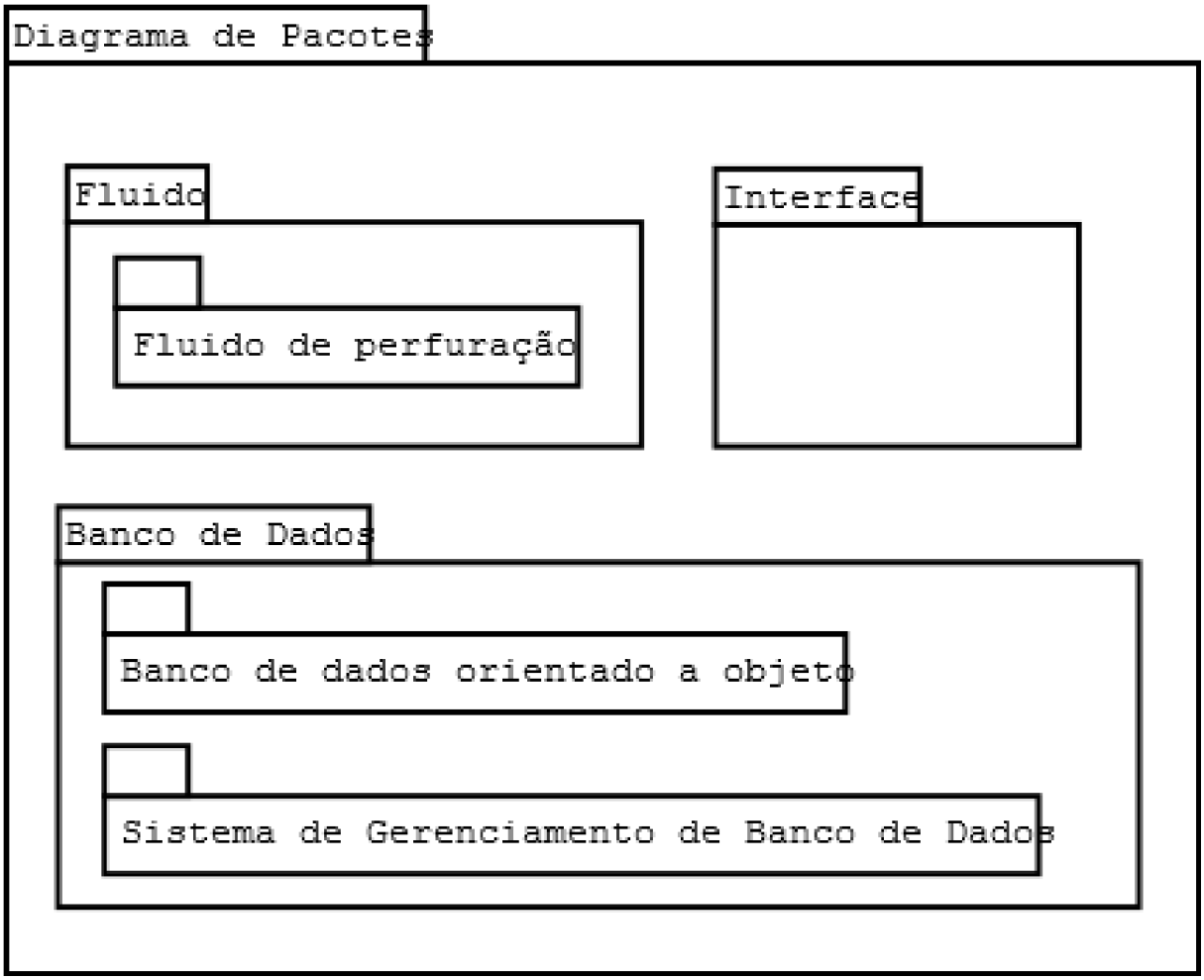


Figura 3.1: Diagrama de pacotes

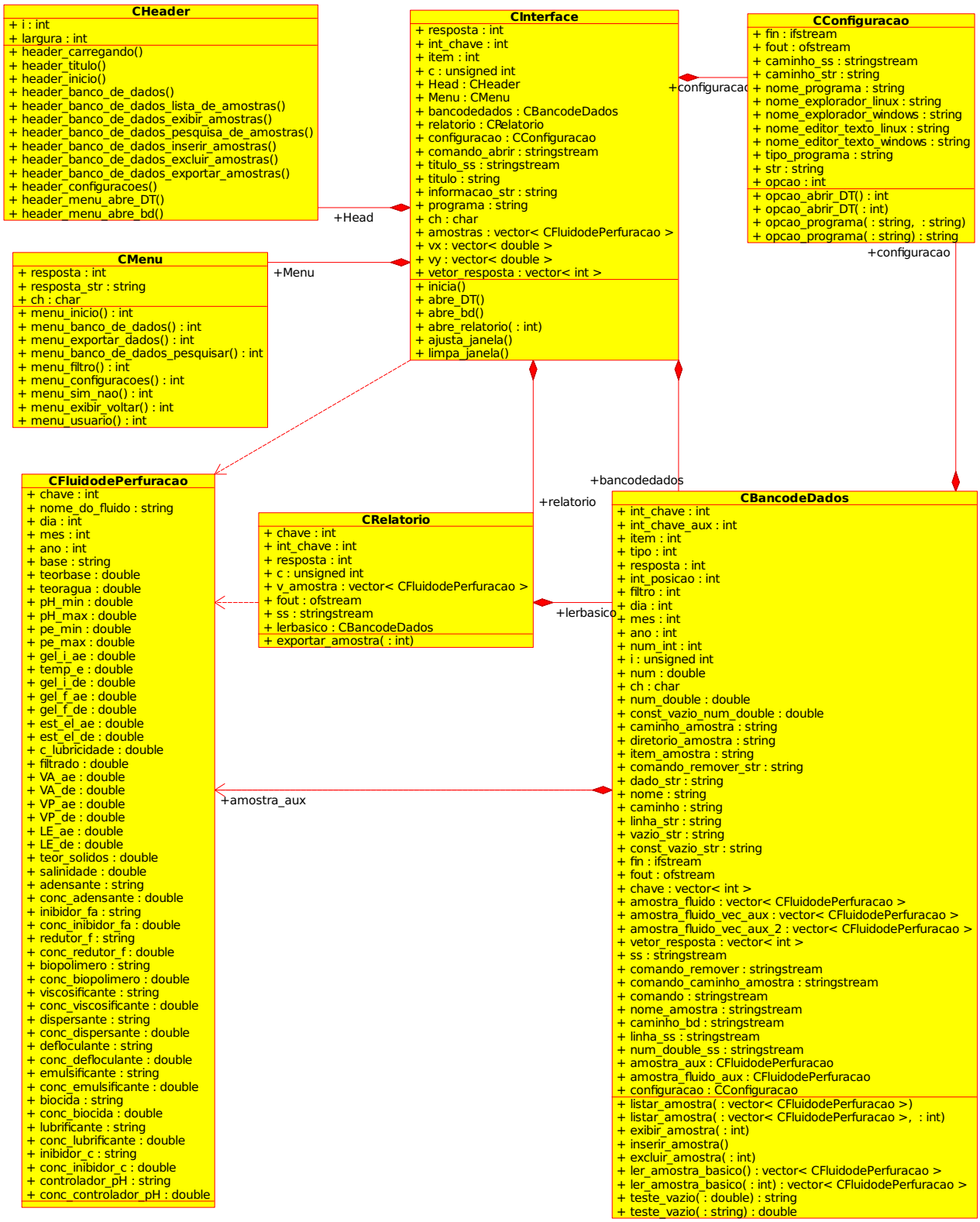
Capítulo 4

AOO – Análise Orientada a Objeto

A terceira etapa do desenvolvimento de um sistema é a Análise Orientada a Objeto (AOO). A AOO utiliza algumas regras para identificar os objetos de interesse, as relações entre os pacotes, as classes, os atributos, os métodos, as heranças, as associações, as agregações, as composições e as dependências. O resultado da análise é um conjunto de diagramas que identificam os objetos e seus relacionamentos.

4.1 Diagramas de Classes

O diagrama de classes é apresentado na Figura 4.1.


CHeader
+ i : int
+ largura : int
+ header_carregando()
+ header_titulo()
+ header_inicio()
+ header_banco_de_dados()
+ header_banco_de_dados_lista_de_amostras()
+ header_banco_de_dados_exibir_amostras()
+ header_banco_de_dados_pesquisa_de_amostras()
+ header_banco_de_dados_inserir_amostras()
+ header_banco_de_dados_excluir_amostras()
+ header_banco_de_dados_exportar_amostras()
+ header_configuracoes()
+ header_menu_abre_DT()
+ header_menu_abre_bd()
+Head
CInterface
+ resposta : int
+ int_chave : int
+ item : int
+ c : unsigned int
+ Head : CHeader
+ Menu : CMenu
+ bancodedados : CBancodeDados
+ relatorio : CRelatorio
+ configuracao : CConfiguracao
+ comando_abrir : stringstream
+ titulo_ss : stringstream
+ titulo : string
+ informacao_str : string
+ programa : string
+ ch : char
+ amostras : vector< CFluidodePerfuracao >
+ vx : vector< double >
+ vy : vector< double >
+ vetor_resposta : vector< int >
+ inicia()
+ abre_DT()
+ abre_bd()
+ abre_relatorio(: int)
+ ajusta_janela()
+ limpa_janela()
+configuracao
CConfiguracao
+ fin : ifstream
+ fout : ofstream
+ caminho_ss : stringstream
+ caminho_str : string
+ nome_programa : string
+ nome_explorador_linux : string
+ nome_explorador_windows : string
+ nome_editor_texto_linux : string
+ nome_editor_texto_windows : string
+ tipo_programa : string
+ str : string
+ opcao : int
+ opcao_abrir_DT() : int
+ opcao_abrir_DT(: int)
+ opcao_programa(: string, : string)
+ opcao_programa(: string) : string
+configuracao
CMenu
+ resposta : int
+ resposta_str : string
+ ch : char
+ menu_inicio() : int
+ menu_banco_de_dados() : int
+ menu_exportar_dados() : int
+ menu_banco_de_dados_pesquisar() : int
+ menu_filtro() : int
+ menu_configuracoes() : int
+ menu_sim_nao() : int
+ menu_exibir_voltar() : int
+ menu_usuario() : int
+Menu
CFluidodePerfuracao
+ chave : int
+ nome_do_fluido : string
+ dia : int
+ mes : int
+ ano : int
+ base : string
+ teorbase : double
+ teoragua : double
+ pH_min : double
+ pH_max : double
+ pe_min : double
+ pe_max : double
+ gel_i_ae : double
+ temp_e : double
+ gel_i_de : double
+ gel_f_ae : double
+ gel_f_de : double
+ est_el_ae : double
+ est_el_de : double
+ c_lubricidade : double
+ filtrado : double
+ VA_ae : double
+ VA_de : double
+ VP_ae : double
+ VP_de : double
+ LE_ae : double
+ LE_de : double
+ teor_solidos : double
+ salinidade : double
+ adensante : string
+ conc_adensante : double
+ inibidor_fa : string
+ conc_inibidor_fa : double
+ redutor_f : string
+ conc_redutor_f : double
+ biopolimero : string
+ conc_biopolimero : double
+ viscosificante : string
+ conc_viscosificante : double
+ dispersante : string
+ conc_dispersante : double
+ defloculante : string
+ conc_defloculante : double
+ emulsificante : string
+ conc_emulsificante : double
+ biocida : string
+ conc_biocida : double
+ lubrificante : string
+ conc_lubrificante : double
+ inibidor_c : string
+ conc_inibidor_c : double
+ controlador_pH : string
+ conc_controlador_pH : double
CRelatorio
+ chave : int
+ int_chave : int
+ resposta : int
+ c : unsigned int
+ v_amostra : vector< CFluidodePerfuracao >
+ fout : ofstream
+ ss : stringstream
+ lerbasico : CBancodeDados
+ exportar_amostra(: int)
+relatorio
+bancodedados
+lerbasico
+amostra_aux
CBancodeDados
+ int_chave : int
+ int_chave_aux : int
+ item : int
+ tipo : int
+ resposta : int
+ int_posicao : int
+ filtro : int
+ dia : int
+ mes : int
+ ano : int
+ num_int : int
+ i : unsigned int
+ num : double
+ ch : char
+ num_double : double
+ const_vazio_num_double : double
+ caminho_amostra : string
+ diretorio_amostra : string
+ item_amostra : string
+ comando_remover_str : string
+ dado_str : string
+ nome : string
+ caminho : string
+ linha_str : string
+ vazio_str : string
+ const_vazio_str : string
+ fin : ifstream
+ fout : ofstream
+ chave : vector< int >
+ amostra_fluido : vector< CFluidodePerfuracao >
+ amostra_fluido_vec_aux : vector< CFluidodePerfuracao >
+ amostra_fluido_vec_aux_2 : vector< CFluidodePerfuracao >
+ vetor_resposta : vector< int >
+ ss : stringstream
+ comando_remover : stringstream
+ comando_caminho_amostra : stringstream
+ comando : stringstream
+ nome_amostra : stringstream
+ caminho_bd : stringstream
+ linha_ss : stringstream
+ num_double_ss : stringstream
+ amostra_aux : CFluidodePerfuracao
+ amostra_fluido_aux : CFluidodePerfuracao
+ configuracao : CConfiguracao
+ listar_amostra(: vector< CFluidodePerfuracao >)
+ listar_amostra(: vector< CFluidodePerfuracao >, : int)
+ exibir_amostra(: int)
+ inserir_amostra()
+ excluir_amostra(: int)
+ ler_amostra_basico() : vector< CFluidodePerfuracao >
+ ler_amostra_basico(: int) : vector< CFluidodePerfuracao >
+ teste_vazio(: double) : string
+ teste_vazio(: string) : double

4.1.1 Dicionário de Classes

- Classe CFluidodePerfuracao: representa a classe que contém informações do fluido de perfuração como seus aditivos, base, propriedades físicas e químicas..
  - atributo chave: representa a chave de identificação do fluido.
  - atributo base: representa o tipo de base que compõe o fluido: água, óleo ou gás.
  - atributo teorbase: representa o teor (%) de base no fluido.
  - atributo teoragua: representa o teor (%) de água no fluido.
  - atributo adensante: representa o adensante usado podendo ser dolomita, por exemplo.
  - atributo conc_adensante: representa a concentração de adensante no fluido de perfuração.
  - atributo inibidor_fa: representa o inibidor de formações ativas (sal) usado podendo ser cloreto de sódio, por exemplo.
  - atributo conc_inibidor_fa: representa a concentração do sal no fluido de perfuração.
  - atributo redutor_f: representa o redutor de filtrado usado podendo ser amido, por exemplo.
  - atributo conc_redutor_f: representa a concentração do redutor de filtrado no fluido de perfuração.
  - atributo biopolimero: representa o biopolímero usado podendo ser carboximetilcelulose, por exemplo.
  - atributo conc_biopolimero: representa a concentração do biopolímero no fluido de perfuração.
  - atributo viscosificante: representa o controlador de viscosidade usado podendo ser bentonita, por exemplo.
  - atributo conc_viscosificante: representa a concentração do sal no fluido de perfuração.
  - atributo dispersante: representa o dispersante usado podendo ser lignosulfonatos, por exemplo.
  - atributo conc_dispersante: representa a concentração do dispersante no fluido de perfuração.
  - atributo defloculante: representa o defloculante usado podendo ser poliacrilato de cálcio, por exemplo.

- atributo conc_defloculante: representa a concentração do defloculante no fluido de perfuração.
- atributo emulsificante: representa o emulsificante usado podendo ser ácido graxo, por exemplo.
- atributo conc_emulsificante: representa a concentração do emulsificante no fluido de perfuração.
- atributo biocida: representa o biocida usado podendo ser glutaraldeído, por exemplo.
- atributo conc_biocida: representa a concentração do biocida no fluido de perfuração.
- atributo lubrificante: representa o lubrificante usado podendo ser ácido graxo, por exemplo.
- atributo conc_lubrificante: representa a concentração do lubrificante no fluido de perfuração.
- atributo inibidor_c: representa o inibidor de corrosão usado podendo ser ácidos de cadeia longa, por exemplo.
- atributo conc_inibidor_c: representa a concentração do inibidor de corrosão no fluido de perfuração.
- atributo controlador_pH: representa o controlador de pH usado podendo ser hidróxido de sódio, por exemplo.
- atributo conc_controlador_pH: representa a concentração do controlador de pH no fluido de perfuração.
- atributo pH_min: representa o valor inferior da faixa de pH para o fluido desenvolvido.
- atributo pH_max: representa o valor superior da faixa de pH para o fluido desenvolvido.
- atributo pe_min: representa o valor inferior da faixa de peso específico para o fluido desenvolvido.
- atributo pe_max: representa o valor superior da faixa de peso específico para o fluido desenvolvido.
- atributo gel_i_ae: representa a força gel, forças atrativas elétricas dentro de um fluido de perfuração, quando submetido às condições estáticas após 10 segundos e antes do envelhecimento;
- atributo temp_e: representa a temperatura de envelhecimento, temperatura da estufa aquecedora na qual o fluido foi inserido numa célula de aço;

  - atributo gel_i_de: representa a força gel, forças atrativas elétricas dentro de um fluido de perfuração, quando submetido às condições estáticas após 10 segundos e depois do envelhecimento;
  - atributo gel_f_ae: representa a força gel, forças atrativas elétricas dentro de um fluido de perfuração, quando submetido às condições estáticas após 10 minutos e antes do envelhecimento;
  - atributo gel_f_de: representa a força gel, forças atrativas elétricas dentro de um fluido de perfuração, quando submetido às condições estáticas após 10 minutos e depois do envelhecimento;
  - atributo est_el_ae: representa a estabilidade elétrica medida pela voltagem requerida para iniciar um fluxo de corrente no fluido antes do envelhecimento.
  - atributo est_el_de: representa a estabilidade elétrica medida pela voltagem requerida para iniciar um fluxo de corrente no fluido depois do envelhecimento.
  - atributo c_lubricidade: representa o coeficiente de lubricidade do fluido de perfuração;
  - atributo filtrado: representa o volume de filtrado obtido em análise laboratorial.
  - atributo VA_ae: representa a viscosidade aparente do fluido antes do envelhecimento.
  - atributo VA_de: representa a viscosidade aparente do fluido depois do envelhecimento.
  - atributo VP_ae: representa a viscosidade plástica do fluido antes do envelhecimento.
  - atributo VP_de: representa a viscosidade plástica do fluido depois do envelhecimento.
  - atributo LE_ae: representa o limite de escoamento do fluido antes do envelhecimento.
  - atributo LE_de: representa o limite de escoamento do fluido depois do envelhecimento.
  - atributo teor_solidos: representa o teor (%) de sólidos presentes no fluido.
  - atributo salinidade: representa a salinidade do fluido.

- Classe CBancodeDados: representa toda estrutura para o desenvolvimento e gerenciamento do banco de dados.

  - atributo int_chave: inteiro que armazena uma linha do arquivo com as chaves.
  - atributo chave_aux: vetor que armazena as chaves lidas no arquivo.

- atributo item: armazena os titulos e subtitulos nos arquivos com as informações dos fluidos.
- atributo tipo:
- atributo resposta: armazena resposta do usuário.
- atributo filtro: representa o filtro a ser utilizado na pesquisa ao banco de dados.
- atributo dia: representa o dia de cadastro da amostra.
- atributo mes: representa o mês de cadasto da amostra.
- atributo ano: representa o ano de cadastro da amostra.
- atributo num_int: representa um número inteiro.
- atributo caminho_fluido: representa o caminho dos arquivos que contém as informações dos fluidos no computador.
- atributo caminho_bd: representa o caminho do banco de dados no computador.
- atributo fluido: vetor de fluidos de perfuração do tipo CFluidodePerfuração.
- atributo fluido_aux: vetor auxiliar;
- atributo i: contador.
- atributo num: representa um numero.
- char ch: representa um caracter.
- atributo num_double:representa um número do tipo double.
- atributo const_vazio_num_double: representa um número do tipo double.
- atributo caminho_amostra: representa o caminho onde se encontra a pasta da amostra de fluido.
- atributo diretorio_amostra: representa o caminho onde se encontra o diretorio.
- atributo nome_amostra: representa o nome da amostra.
- atributo item_amostra: faz a leitura dos títulos do arquivo .
- atributo comando_remover_str: comando para remover amostra em string.
- atributo nome: representa o nome dado pelo usuário para a amostra.
- atributo caminho: representa o caminho que se encontra a pasta da amostra de fluido.
- atributo linha_str: representa a leitura de uma linha do arquivo.
- atributo fin: variavel de leitura do arquivo.
- atributo fout: variavel de escrita do arquivo.
- atributo chave: Vetor de chaves lidas no arquivo.

  - atributo amostra_fluido: vetor de amostras
  - atributo amostra_fluido_vec_aux: vetor de amostras auxiliar.
  - atributo amostra_fluido_vec_aux_2: vetor de amostras auxiliar.
  - atributo vetor_resposta: vetor que armazena resposta.
  - atributo ss: stringstream para conversao de inteiro para string stringstream.
  - atributo comando_remover: cria comando para remover amostra para lixeira.
  - atributo comando_caminho_amostra: cria comando para criar um caminho para o diretório.
  - atributo comando: cria comando para criar o diretório.
  - atributo caminho_bd: cria comando para criar um caminho para a pasta bd.
  - atributo linha_ss: para ler a linha do arquivo.
  - atributo num_double_ss: converte o double para string.
  - método Listar_fluido(vector<CFluidodePerfuracao>): Lista os fluidos de perfuração.
  - método Listar_fluido(vector<CFluidodePerfuracao>,int): Lista fluidos de acordo com um filtro.
  - método Exibir_fluido(int chave_): Exibe um determinado fluido.
  - método Inserir_fluido(): Insere um fluido ao banco de dados.
  - método Excluir_amostra(int): Exclui um fluido do banco de dados.
  - método Ler_fluido_basico(): Realiza a leitura dos arquivos referentes aos fluidos.
  - método ler_amostra_basico(int): : Realiza a leitura dos arquivos referentes aos fluidos filtrados.
  - método ler_informacao(int): Ler informações dos fluidos.
  - método teste_vazio(double): para correções de respostas do usuário e conversões.
  - método teste_vazio(string): para correções de respostas do usuário e conversões.

- Classe CInterface: representa a interface em modo texto do programa.

  - atributo resposta: representa a resposta ao usuário referente ao menu de opções.
  - atributo item: leitura do título do arquivo.
  - atributo c: contador.

  - atributo comando_abrir: comando para abrir um diretório.
  - atributo titulo: representa os títulos que aparecerão para o usuário ao selecionar uma opção do menu.
  - atributo informacao_str:informação da amostra.
  - atributo programa: recebe um comando para configurar programa.
  - atributo senha: senha do administrador para configurar o programa.
  - atributo senha_confirma: senha a ser confirmada inserida pelo usuário.
  - atributo ch: caracter.
  - atributo amostras: vetor do tipo vector<CFluidodePerfuracao> de amostras.
  - atributo vetor_resposta: vetor do tipo vector<int> que armazena as respostas.
  - método inicia(): método para iniciar o programa.
  - método abre_DT(): método para abrir diretório de trabalho.
  - método abre_bd():método para abrir pasta do banco de dados.
  - método abre_relatorio(int): método para abrir relatórios.
  - método ajusta_janela(): método para ajustar janela.
  - método limpa_janela(): método para limpar janela.
  - método gerenciar_usuario(CUsuario&): método para gerenciar usuário (administrador e usuário comum).

- Classe CMenu: representa a classe com todos menus.

  - atributo resposta: representa a resposta do usuário.
  - atributo resposta_str: representa a resposta do usuário convertida para string.
  - atributo ch: representa um caracter.
  - atributo vetor_resposta: vetor do tipo vector<int> que armazena a resposta do usuário..
  - método menu_inicio(): método referente ao menu de início.
  - método menu_banco_de_dados(): método referente ao menu que lista as funções referentes ao banco de dados.
  - método menu_exportar_dados(): menu referente à geração de relatórios.
  - método menu_banco_de_dados_pesquisar(): menu referente às opções de pesquisa.
  - método menu_filtro(): menu referente às opções de filtro.
  - método menu_configuracoes(): menu referente às opções de configurações.

  - método menu_sim_nao(): menu referente à resposta do usuário (sim ou não).
  - método menu_exibir_voltar(): menu referente às opções exibir ou voltar.
  - método menu_usuario(): menu oculto referente ao gerenciamento de usuário.

- Classe CRelatorio: representa a classe que exporta as amostras em forma de relatório.
  - atributo chave: representa a chave da amostra a ser exportada.
  - atributo int_chave: representa a chave da amostra a ser exportada.
  - atributo resposta: representa a resposta do usuário.
  - atributo c: contador.
  - atributo v_amostra: vetor do tipo vector<CFluidodePerfuracao> de amostra.
  - método exportar_amostra(int): método para exportar as amostras em forma de relatório.

- CConfiguracao: representa a classe que configura o explorador e o editor de texto.
  - atributo caminho_ss: representa o caminho que vai receber a opção do usuário para abrir sempre ou não os diretórios.
  - atributo nome_programa: nome do programa definido pelo usuário.
  - atributo nome_explorador_linux: representa o explorador do GNU/linux.
  - atributo nome_explorador_windows: representa o explorador do windows.
  - atributo nome_editor_texto_linux: representa o nome do editor texto do GNU/linux.
  - atributo nome_editor_texto_windows: representa o nome do editor texto do windows.
  - atributo tipo_programa: representa o tipo de programa.
  - atributo opcao: representa a opção selecionada pelo usuário no menu.
  - método opcao_abrir_DT(): método para abrir diretório.
  - método opcao_abrir_DT(int): método para abrir diretório.
  - método opcao_programa(string, string): método para a troca do explorador e do editor de texto.
  - método opcao_programa(string): método para a troca do explorador

- CHeader: representa a classe que implementa os cabeçalhos.
  - atributo status_usuario: representa o status que aparecerá na tela de acordo com o tipo de usuário.

- atributo i: contador.
- atributo largura: vetor de largura do retângulo do titulo.
- método header_carregando(): referente à animação de carregamento do programa.
- método header_titulo(int): escreve titulo do programa.
- método header_inicio(): escreve subtítulo do programa.
- método header_banco_de_dados(): escreve subtítulo de listar amostras.
- método header_banco_de_dados_lista_de_amostras(): escreve subtítulo de amostras listadas.
- método header_banco_de_dados_exibir_amostras(): escreve subtitulo de exibir amostras.
- método header_banco_de_dados_pesquisa_de_amostras(): escreve subtitulo de amostras listadas.
- método header_banco_de_dados_inserir_amostras(): escreve subtitulo de inserir amostras.
- método header_banco_de_dados_excluir_amostras(): escreve subtitulo de excluir amostras.
- método header_banco_de_dados_exportar_amostras(): escreve subtitulo de excluir amostras.
- método header_banco_de_dados_informacao_amostras(): escreve subtitulo de informações de amostras.
- método header_configuracoes(): escreve subtitulo de configurações.
- método header_menu_abre_DT(): escreve subtitulo do menu abrir diretório.
- método header_menu_abre_bd(): escreve subtitulo do menu banco de dados.
- método header_tipo_usuario(): escreve subtitulo do menu.

4.2 Diagrama de Sequência

O diagrama de sequência enfatiza a troca de eventos e mensagens e sua ordem temporal. Contém informações sobre o fluxo de controle do programa. Estabelece o relacionamento dos atores (usuários e sistemas externos) com alguns objetos do sistema.

Veja o diagrama de sequência na Figura 4.2.

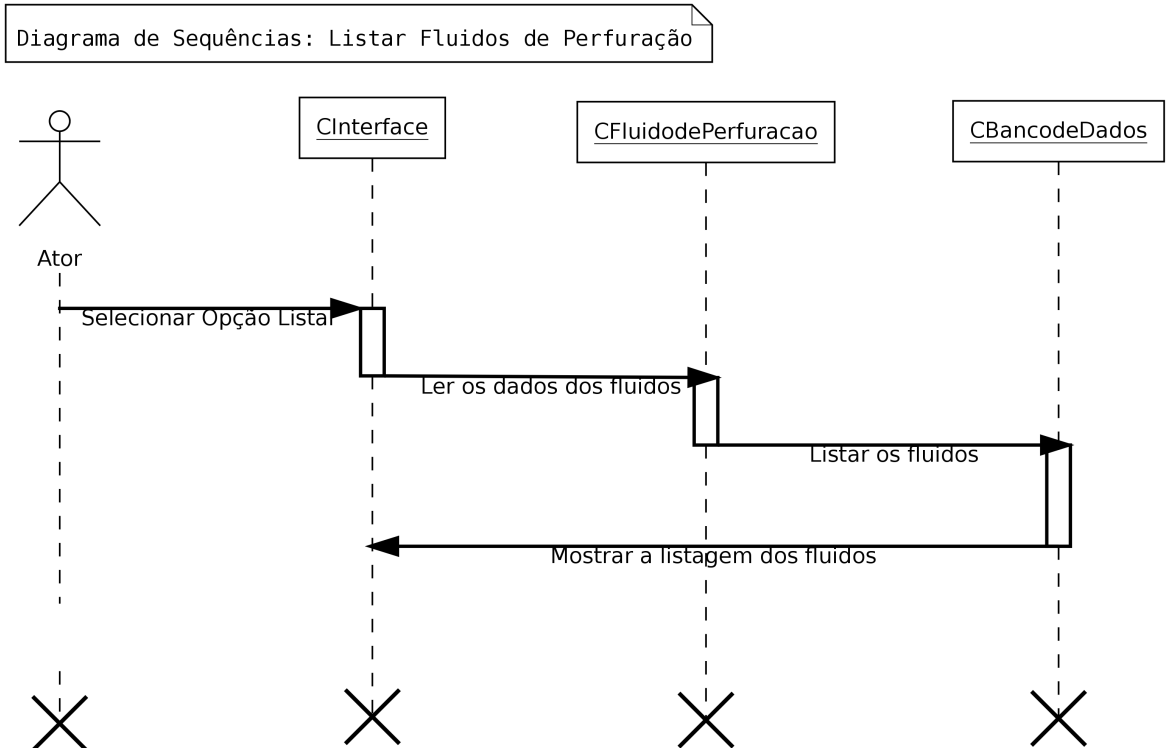


Figura 4.2: Diagrama de seqüência: Listar Fluidos de Perfuração

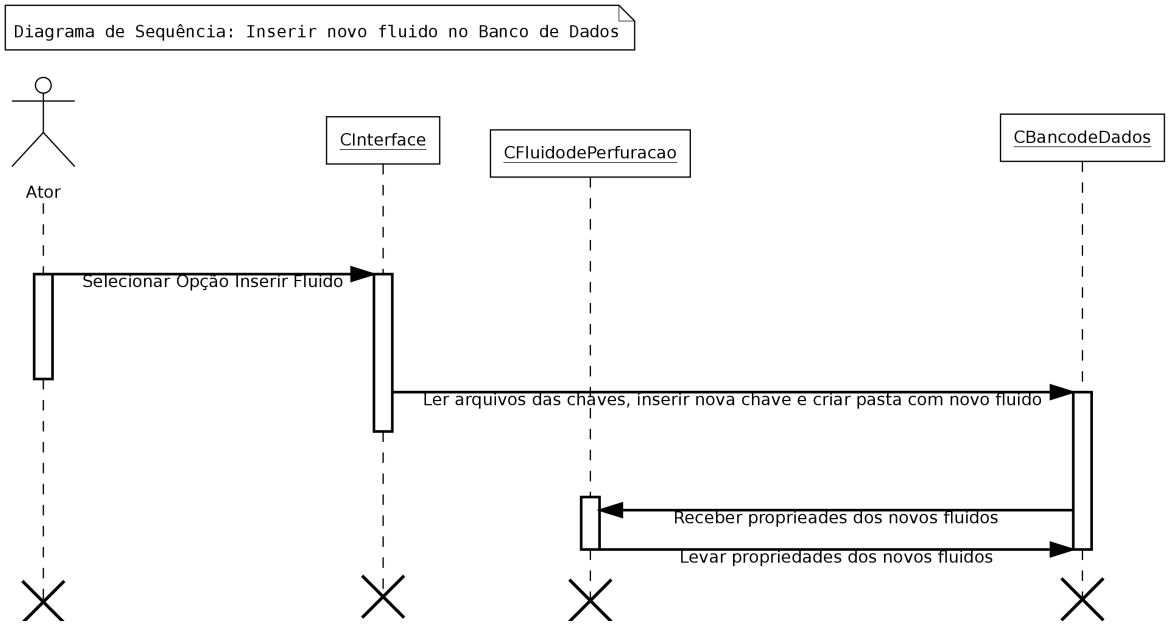


Figura 4.3: Diagrama de Sequência: Pesquisar Fluidos com propriedades específicas

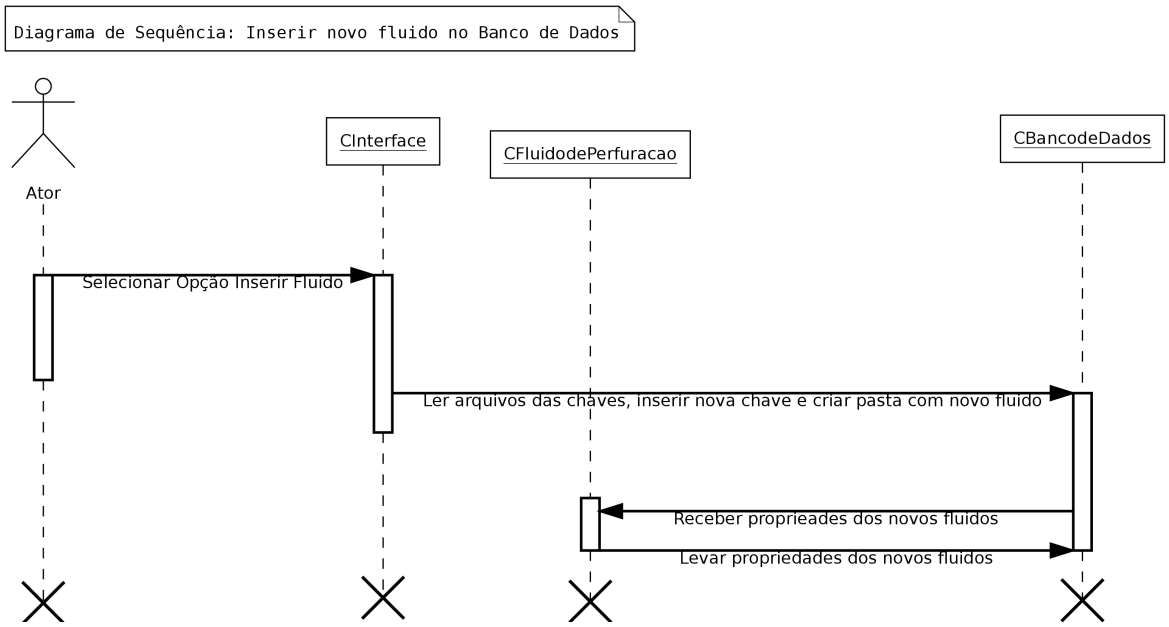


Figura 4.4: Diagrama se Sequência: Inserir Novo Fluido ao banco de Dados

4.3 Diagrama de Colaboração

O diagrama de colaboração pode ser desenvolvido como uma extensão do diagrama de caso de uso, detalhando o mesmo por meio da inclusão de objetos, mensagens e parâmetros trocados entre objetos. O principal objetivo deste diagrama é a interação e a troca de mensagens e dados entre os objetos.

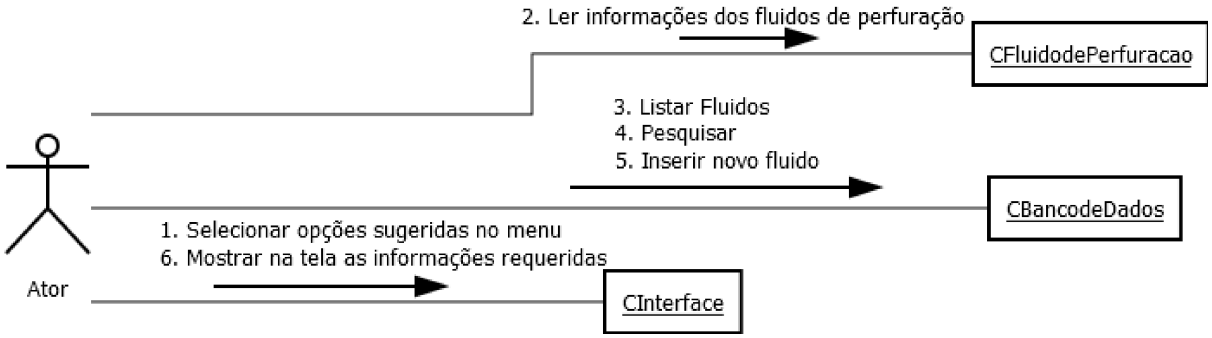


Figura 4.5: Diagrama de colaboração: acessando o banco de dados.

4.4 Diagrama de Máquina de Estado

Um diagrama de máquina de estado representa os diversos estados que o objeto assume e os eventos que ocorrem ao longo de sua vida ou mesmo ao longo de um processo (histórico do objeto). É usado para modelar aspectos dinâmicos do objeto. Existem basicamente dois usos para máquinas de estado: máquinas de estado comportamentais e máquinas de estado para protocolos.

Máquinas de estado comportamentais podem ser utilizadas para especificar o comportamento de vários tipos de elementos. Por exemplo, podem ser utilizadas para modelar o comportamento de entidades individuais (objetos), por meio da modificação dos valores de seus atributos.

Máquinas de estado para protocolos expressam as transições legais que um objeto pode desenvolver. Com seu uso, pode-se definir o ciclo de vida de objetos, ou uma determina ordem na invocação de suas operações. Para este tipo de máquina de estado, interfaces e portas podem estar associados.

Veja na Figura 4.6 o diagrama de máquina de estado para o banco de dados.

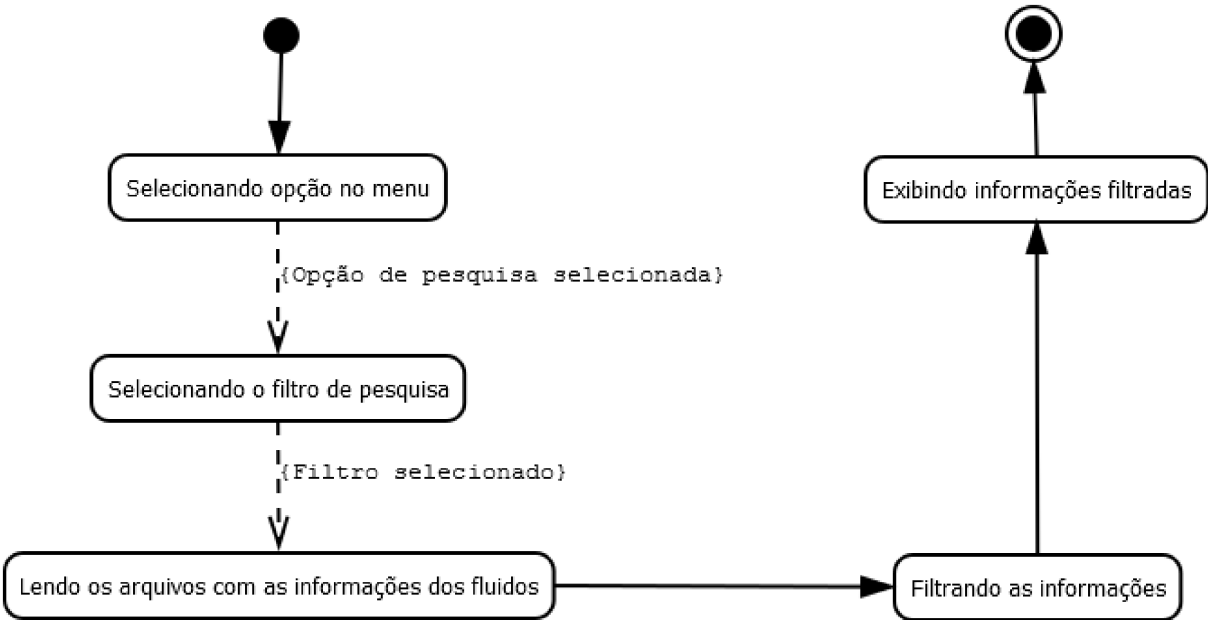


Figura 4.6: Diagrama de máquina de estado: classe CBancodeDados

4.5 Diagrama de Atividades

O diagrama de atividades é um diagrama UML utilizado para modelar o aspecto comportamental de processos. Neste diagrama, uma atividade é modelada como uma sequência estruturada de ações controladas por nós de decisão e sincronismo.

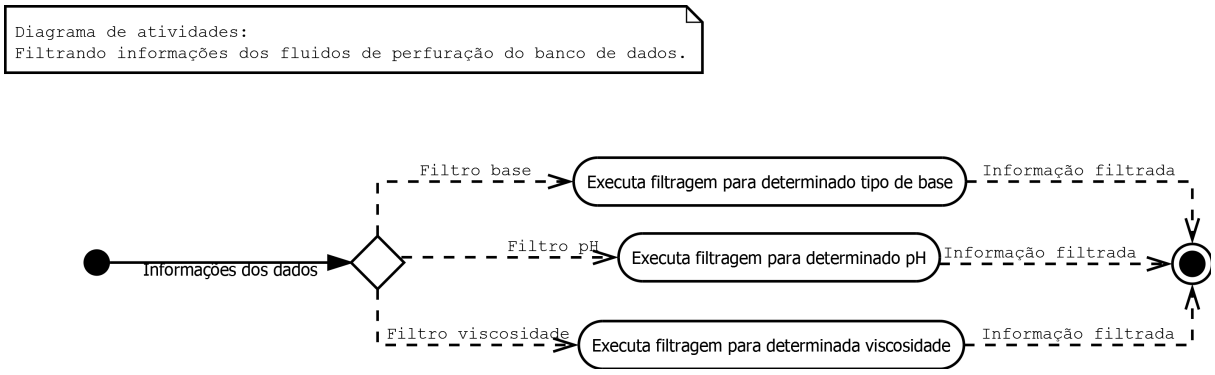


Figura 4.7: Diagrama de atividades: Classe CBancodeDados: listar_amostra(vector<CFluidodePerfuracao>, int).

Capítulo 5

Projeto

Depois da análise orientada a objeto desenvolve-se o projeto do sistema, o qual envolve etapas como a definição dos protocolos, da interface API, o uso de recursos, a subdivisão do sistema em subsistemas, a alocação dos subsistemas ao hardware e a seleção das estruturas de controle, a seleção das plataformas do sistema, das biblitoecas externas, dos padrões de projeto, além da tomada de decisões conceituais e políticas que formam a infraestrutura do projeto.

Deve-se definir padrões de documentação, padrões para o nome das classes, padrões de retorno e de parâmetros em métodos, características da interface do programa e características de desempenho.

5.1 Projeto do Sistema

Segundo Rumbaugh et al. 1994, o projeto do sistema é a estratégia de alto nível para resolver o problema e elaborar uma solução. Deve-se preocupar com itens como:

• Definição do protocolo de comunicação entre os diversos elementos externos (como dispositivos);

• Definição de loops de controle, das escalas de tempo;

• Identificação de subsistemas;

• Identificação de concorrências;

• Identificação de depósitos de dados (implicam em modicações no diagrama de atividades);

• Identificação e alocação dos recursos globais, das condições extremas e de prioridades (implicam em modicações no diagrama de componentes);

• Identificação e seleção da implementação de controle (implicam em modicações no diagrama de implantação);

• Identificação das estruturas arquitetônicas comuns.

1. Protocolos

- Definição dos protocolos de comunicação entre os diversos elementos externos.
  - Esta versão do software não inclui comunicação com elementos externos (internet).
- Definição dos protocolos de comunicação entre os diversos elementos internos.
  - O programa utilizará uma máquina computacional com HD, processador, teclado para a entrada de dados e o monitor para a saída de dados. Os arquivos gerados pelo programa estarão em formato de texto em um banco de dados.
- Definição do formato dos arquivos gerados pelo programa.
  - Os arquivos de texto serão gerados em formato ASCII (não formatado).

2. Recursos

- Identificação e alocação dos recursos globais, como os recursos do sistema serão alocados, utilizados, compartilhados e liberados. Implicam modificações no diagrama de componentes.
  - Não será considerado nessa versão.
- Identificação da necessidade do uso de banco de dados. Implicam em modificações nos diagramas de atividades e de componentes.
  - O programa implementará um banco de dados.

3. Controle

- Identificação da necessidade de otimização.
  - Não será considerado nessa versão.
- Identificação e definição de *loops* de controle e das escalas de tempo.
  - Não será considerado nessa versão.
- Identificação de concorrências.
  - Não será considerado nessa versão.

4. Plataformas

- Identificação das estruturas arquitetônicas comuns.
  - O programa será desenvolvido em linguagem C++ usando o conceito de programação orientada a objeto.
- Identificação e definição das plataformas a serem suportadas: hardware, sistema operacional e linguagem de programação.

  - O programa será desenvolvido em computador Intel 32/64 bits, com sistema operacional Windows usando linguagem C++ orientada a objeto.
- Seleção das bibliotecas externas
  - Não será considerado nessa versão.

5. Subsistemas

- Identificação dos subsistemas:
  - Fluido de Perfuração;
  - Perfuração;
  - Banco de Dados;

5.2 Projeto Orientado a Objeto – POO

O projeto orientado a objeto é a etapa posterior ao projeto do sistema. Baseia-se na análise, mas considera as decisões do projeto do sistema. Acrescenta a análise desenvolvida e as características da plataforma escolhida (hardware, sistema operacional e linguagem de programação). Passa pelo maior detalhamento do funcionamento do programa, acrescentando atributos e métodos que envolvem a solução de problemas específicos não identificados durante a análise.

Envolve a otimização da estrutura de dados e dos algoritmos, a minimização do tempo de execução, de memória e de custos. Existe um desvio de ênfase para os conceitos da plataforma selecionada.

Efeitos do projeto no modelo estrutural

- Novas classes e associações oriundas das bibliotecas selecionadas e da linguagem escolhida devem ser acrescentadas ao modelo.
  - Após a análise e o projeto do sistema surgiu a necessidade da criação de novas classes e associações. Problemas como esse poderão surgir durante a implementação do banco de dados, sendo assim passível de modificação ou criação de novas classes, atributos e métodos.
- Estabelecer as dependências e restrições associadas à plataforma escolhida.

Efeitos do projeto no modelo dinâmico

- Revisar os diagramas de sequência e de colaboração considerando a plataforma escolhida.

  - Não houve alteração.

- Verificar a necessidade de se revisar, ampliar e adicionar novos diagramas de máquina de estado e de atividades.

  - Não houve alteração.

Efeitos do projeto nos atributos

- Atributos novos podem ser adicionados a uma classe, como, por exemplo, atributos específicos de uma determinada linguagem de programação (acesso a disco, ponteiros, constantes e informações correlacionadas).

  - Não houve alteração.

Efeitos do projeto nas heranças

- Reorganização da herança no diagrama de classes.

  - Não houve alteração.

Efeitos do projeto nas associações

- Reorganização das associações.

  - Não houve alteração.

Efeitos do projeto nas otimizações

- Otimização do sistema.

  - Não houve alteração.

- Identifique pontos a serem otimizados em que podem ser utilizados processos concorrentes.

  - Não houve alteração.

5.3 Diagrama de Componentes

O diagrama de componentes mostra a forma como os componentes do programa se relacionam, suas dependências. Inclui itens como: componentes, subsistemas, executáveis, nós, associações, dependências, generalizações, restrições e notas. Exemplos de componentes são: Bibliotecas estáticas, bibliotecas dinâmicas, dlls, componentes Java, executáveis, arquivos de disco e código-fonte.

5.4 Diagrama de Execução

O diagrama de implantação (execução) é um diagrama de alto nível que inclui relações entre o sistema e o hardware e que se preocupa com os aspectos da arquitetura computacional escolhida. Seu enfoque é o hardware.

O diagrama de implantação deve incluir os elementos necessários para que o sistema seja colocado em funcionamento: computador, periféricos, processadores, dispositivos, nós, relacionamentos de dependência, associação, componentes, subsistemas, restrições e notas.

Para o banco de dados será necessário apenas um computador que possua HD, teclado para a entrada de dados e monitor para a saída de dados. Também não será preciso uma rede de computadores, visto que o banco de dados depende somente da interação usuário - simulador ((Figura 5.1).

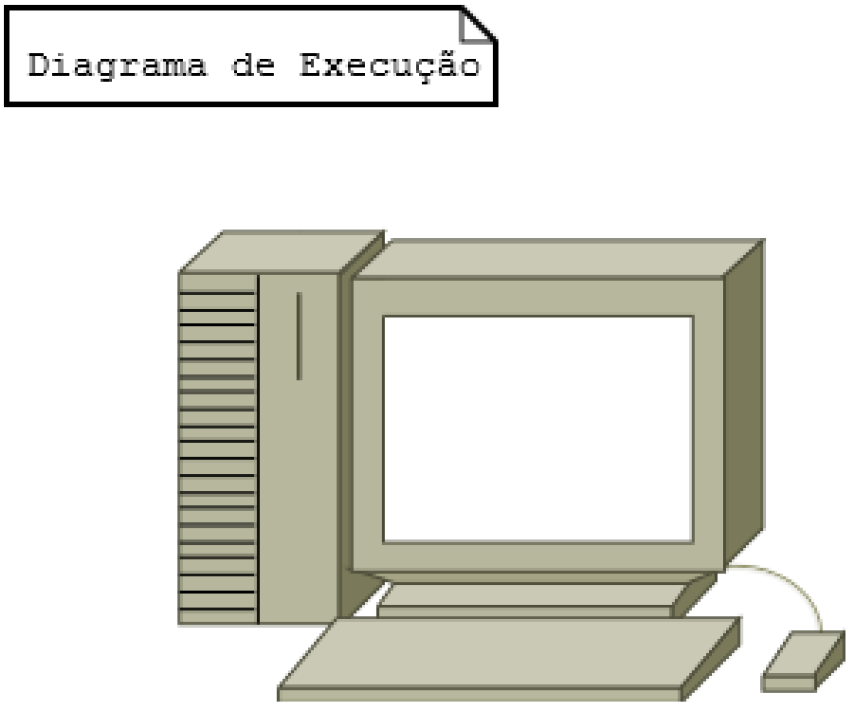


Figura 5.1: Diagrama de Execução.

Capítulo 6

Implementação

Neste capítulo do projeto de engenharia apresentamos os códigos fonte que foram desenvolvidos.

6.1 Código Fonte

Apresenta-se a seguir um conjunto de classes (arquivos .h e .cpp) além do programa `main`.

Apresenta-se na listagem 6.1 o arquivo com código da classe CFluidodePerfuracao.

Listing 6.1: Arquivo de cabeçalho da classe CFluidodePerfuracao.

```
1#ifndef CFluidodePerfuracao_h
2#define CFluidodePerfuracao_h
3#include <string>
4#include <vector>
5#include <iostream>
6using namespace std;
7
8class CFluidodePerfuracao
9{
10public:
11  //Atributos
12
13  //Componentes
14  int chave; ///< representa a chave de identificação do fluido.
15  string nome_do_fluido;
16  int dia;
17  int mes;
18  int ano;
19  /// representa o tipo de base que compõe o fluido, água,óleo ou gás.
20  string base;
21  /// representa o teor(%) de base no fluido.
22  double teorbase;
23  /// representa o teor(%) de água no fluido.
```

```
24  double teoragua;
25
26  /// Propriedades fisicas e quimicas
27  /// Representa o valor inferior da faixa de pH para o fluido
        desenvolvido.
28  double pH_min;
29  /// representa o valor superior da faixa de pH para o fluido
        desenvolvido.
30  double pH_max;
31  double pe_min; ///<representa o valor inferior da faixa de peso
        específico para o fluido desenvolvido.
32  double pe_max; ///<representa o valor superior da faixa de peso
        específico para o fluido desenvolvido.
33  double gel_i_ae; ///<representa a força gel, forças atrativas
        elétricas dentro de um fluido de perfuração, quando submetido às
        condições estáticas após 10 segundos e antes do envelhecimento;
34  double temp_e; ///<representa a temperatura de envelhecimento,
        temperatura da estufa aquecedora na qual o fluido foi inserido numa
         célula de aço;
35  double gel_i_de; ///<representa a força gel, forças atrativas
        elétricas dentro de um fluido de perfuração, quando submetido às
        condições estáticas após 10 segundos e depois do envelhecimento;
36  double gel_f_ae; ///<representa a força gel, forças atrativas
        elétricas dentro de um fluido de perfuração, quando submetido às
        condições estáticas após 10 minutos e antes do envelhecimento;
37  double gel_f_de; ///<representa a força gel, forças atrativas
        elétricas dentro de um fluido de perfuração, quando submetido às
        condições estáticas após 10 minutos e depois do envelhecimento;
38  double est_el_ae; ///<representa a estabilidade elétrica medida pela
        voltagem requerida para iniciar um fluxo de corrente no fluido
        antes do envelhecimento.
39  double est_el_de; ///<representa a estabilidade elétrica medida pela
        voltagem requerida para iniciar um fluxo de corrente no fluido
        depois do envelhecimento.
40  double c_lubricidade; ///<representa o coeficiente de lubricidade do
        fluido de perfuração;
41  double filtrado; ///<representa o volume de filtrado obtido em análise
         laboratorial.
42  double VA_ae; ///<representa a viscosidade aparente do fluido antes do
         envelhecimento.
43  double VA_de; ///<representa a viscosidade aparente do fluido depois
        do envelhecimento.
44  double VP_ae; ///<representa a viscosidade plástica do fluido antes do
         envelhecimento.
45  double VP_de; ///<representa a viscosidade plástica do fluido depois
        do envelhecimento.
46  double LE_ae; ///<representa o limite de escoamento do fluido antes do
         envelhecimento.
```

```
47  double LE_de; ///<representa o limite de escoamento do fluido depois
        do envelhecimento.
48  double teor_solidos; ///<representa o teor(%) de sólidos presentes no
        fluido.
49  double salinidade; ///<representa a salinidade do fluido.
50
51  //ADiTIVOS
52  string adensante; ///<representa o adensante usado podendo ser
        dolomita, por exemplo.
53  double conc_adensante; ///<representa a concentração de adensante no
        fluido de perfuração.
54  string inibidor_fa; ///<representa o inibidor de formações ativas (sal
        ) usado podendo ser cloreto de sódio, por exemplo.
55  double conc_inibidor_fa; ///<representa a concentração do sal no
        fluido de perfuração.
56  string redutor_f; ///<representa o redutor de filtrado usado podendo
        ser amido, por exemplo.
57  double conc_redutor_f; ///<representa a concentração do redutor de
        filtrado no fluido de perfuração.
58  string biopolimero; ///<representa o biopolímero usado podendo ser
        carboximetilcelulose, por exemplo.
59  double conc_biopolimero; ///<representa a concentração do biopolímero
        no fluido de perfuração.
60  string viscosificante; ///<representa o controlador de viscosidade
        usado podendo ser bentonita, por exemplo.
61  double conc_viscosificante; ///<representa a concentração do sal no
        fluido de perfuração.
62  string dispersante; ///<representa o dispersante usado podendo ser
        lignosulfonatos, por exemplo.
63  double conc_dispersante; ///<representa a concentração do dispersante
        no fluido de perfuração.
64  string defloculante; ///<representa o defloculante usado podendo ser
        poliacrilato de cálcio, por exemplo.
65  double conc_defloculante; ///<representa a concentração do
        defloculante no fluido de perfuração.
66  string emulsificante; ///<representa o emulsificante usado podendo ser
         ácido graxo, por exemplo.
67  double conc_emulsificante; ///<representa a concentração do
        emulsificante no fluido de perfuração.
68  string biocida; ///<representa o biocida usado podendo ser
        glutaraldeído, por exemplo.
69  double conc_biocida; ///<representa a concentração do biocida no
        fluido de perfuração.
70  string lubrificante; ///<representa o lubrificante usado podendo ser
        ácido graxo, por exemplo.
71  double conc_lubrificante; ///<representa a concentração do
        lubrificante no fluido de perfuração.
72  string inibidor_c; ///<representa o inibidor de corrosão usado podendo
```

```
        ser ácidos de cadeia longa, por exemplo.
73  double conc_inibidor_c; ///<representa a concentração do inibidor de
        corrosão no fluido de perfuração.
74  string controlador_pH; ///<representa o controlador de pH usado
        podendo ser hidróxido de sódio, por exemplo.
75  double conc_controlador_pH; ///<representa a concentração do
        controlador de pH no fluido de perfuração.
76
77};
78
79#endif
```

Apresenta-se na listagem 6.2 o arquivo de implementação da classe CFluidodePerfuracao.

Listing 6.2: Arquivo de implementação da classe CFluidodePerfuracao.

```
80#include <iostream>
81#include <stdio.h>
82///<verifica sistema operacional
83#ifdef __linux
84
85#include <cstdlib>
86
87#elif _WIN32
88
89#include <windows.h>
90#include <stdlib.h>///<para system
91#include <conio.h>
92
93#else
94
95#endif
96
97#include <string>
98#include "CFluidodePerfuracao.h"
```

Apresenta-se na listagem 6.3 o arquivo com código da classe CBancodeDados.

Listing 6.3: Arquivo de cabeçalho da classe CBancodeDados.

```
100#ifndef CBancodeDados_h
101#define CBancodeDados_h
102#include <iostream>
103#include <fstream>
104#include <vector>
105#include <string>
106#include <sstream>///<para conversao de int para string
107#include "CHeader.h"
108#include "CConfiguracao.h"
109#include "CFluidodePerfuracao.h"
```

```
110
111 using namespace std;
112
113 class CBancodeDados
114 {
115
116 public:
117   int int_chave;
118   int int_chave_aux;
119   int item;
120   int tipo;
121   int resposta;///<respsota
122   int int_posicao;
123   int filtro;///<filtro da pesquisa
124   int dia;
125   int mes;
126   int ano;
127   int num_int;
128
129
130   unsigned int i;///< i para contagem do for;
131   double num;
132   char ch;
133   double num_double;
134   double const_vazio_num_double;
135
136   string caminho_amostra;
137   string diretorio_amostra;      ///< string nome_amostra;
138   string item_amostra;
139   string comando_remover_str;///<comando para remover amostra em string
140   string dado_str;
141   string nome;
142   string caminho;
143   string linha_str;
144   string vazio_str;
145   string const_vazio_str;
146
147   ifstream fin;///<variavel de leitura do arquivo
148   ofstream fout;
149
150   vector<int> chave;///<Vetor de chaves lidas no arquivo
151   vector<CFluidodePerfuracao> amostra_fluido; ///<vetor de amostras
152   vector<CFluidodePerfuracao> amostra_fluido_vec_aux;
153   vector<CFluidodePerfuracao> amostra_fluido_vec_aux_2;
154   vector<int> vetor_resposta;
155
156   stringstream ss;///<stringstream para conversao de inteiro para string
157   stringstream comando_remover;///<cria comando para remover amostra
```

```
        para lixeira
158   stringstream comando_caminho_amostra;
159   stringstream comando;
160   stringstream nome_amostra;
161   stringstream caminho_bd;
162   stringstream linha_ss;
163   stringstream num_double_ss;
164
165   CFluidodePerfuracao amostra_aux;
166   CFluidodePerfuracao amostra_fluido_aux;///<obj auxiliar
167   CConfiguracao configuracao;
168
169
170 public:
171   void listar_amostra(vector<CFluidodePerfuracao>);///<lista todas as
        amostras
172   void listar_amostra(vector<CFluidodePerfuracao>, int);
173   void exibir_amostra(int);
174   void inserir_amostra();///<insere amostras
175   void excluir_amostra(int);///<exclui amostras
176   vector<CFluidodePerfuracao> ler_amostra_basico();
177   vector<CFluidodePerfuracao> ler_amostra_basico(int);
178   string teste_vazio(double);
179   double teste_vazio(string);
180
181 };
182
183 #endif
```

Apresenta-se na listagem 6.4 o arquivo de implementação da classe CBancodeDados.

Listing 6.4: Arquivo de implementação da classe CBancodeDados.

```
185 #ifdef __linux
186 #elif _WIN32
187 #include <windows.h>
188 #else
189 #endif
190
191 #include <cstdlib>
192 #include <stdlib.h>///para system
193 #include <stdio.h>
194 #include <string>
195 #include <sstream>///para conversao de int para string
196 #include <vector>
197 #include <fstream>
198 #include <iostream>
199 #include <cstring>
200 #include "CBancodeDados.h"
201 #include "CFluidodePerfuracao.h"
```

```
202
203 using namespace std;
204
205 vector<CFluidodePerfuracao> CBancodeDados::ler_amostra_basico()
206 {
207 //
      ////////////////////////////////////////////////////////////////////////////
208 //Inicio da leitura das chaves (Primeiro le o arquivo de informacao do
      banco de dados)
209 //
      ////////////////////////////////////////////////////////////////////////////
210   vector<int> chave;///<vetor que armazena as chaves
211   vector<CFluidodePerfuracao> amostra_fluido;
212   ifstream fin;///<objeto de leitura
213   int int_chave;///<inteiro que recebe cada linha lida no arquivo
214   unsigned int i;///<inteiro para contagem no for
215
216   #ifdef __linux
217   caminho_bd<<"bd"<<char(47)<<"ini"<<char(47)<<"bdinfo";
218   #elif _WIN32
219   caminho_bd<<"bd"<<char(92)<<"ini"<<char(92)<<"bdinfo";
220   #else
221
222   #endif
223
224   fin.open(caminho_bd.str().c_str());///<abre o arquivo de informacoes
        do bd
225   caminho_bd.str("");
226   while(! fin.eof())     ///<carrega o arquivo de informacoes do bd
227   {
228     fin >> int_chave;
229     chave.push_back(int_chave);
230   }
231   fin.close();
232
233 ////////////////////////////////////////////
234 //Termino da leitura das chaves
235 ////////////////////////////////////////////
236
237 ////////////////////////////////////////////
238 //Inicio da leitura das amostras
239 ////////////////////////////////////////////
240
241   CFluidodePerfuracao amostra_fluido_aux;///<objeto amostra fluido
        auxiliar
242
```

```
243  string caminho_amostra;///<caminho da amostra a ser lida incluindo o
         nome dela
244  string item_amostra;///<string para ler os titulos e subtitulos dos
         respectivos itens das amostras
245  stringstream ss;///<stringstream para conversao de inteiro para string
246  num_int=0;///<inicia variavel auxiliar
247  i=0;
248
249  for(i=0;i<chave.size();i++)///<For para leitura de todas as amostras
250  {
251
252    #ifdef __linux
253    ss<<"bd"<<char(47)<<chave[i]<<char(47)<<chave[i];///<"bd/chave/chave
         " e exatamente assim que o fin le o arquivo(linux)
254    #elif _WIN32
255    ss<<"bd"<<char(92)<<chave[i]<<char(92)<<chave[i];///<"bd\chave\chave
         " e exatamente assim que o fin le o arquivo(windows)
256    #else
257
258    #endif
259    caminho_amostra=ss.str();///<converte o caminho da amostra incluindo
          o nome para string
260    ss.str("");///<apaga conteudo de ss
261    fin.open(caminho_amostra.c_str());///<abre todas as amostras
         existentes
262
263    //
         ////////////////////////////////////////////////////////////////////////
264    //lÃª o arquivo de texto e atribui os dados do respectivo objeto
         amostra_fluido[i]////
265    //
         ////////////////////////////////////////////////////////////////////////
266    //Nesta etapa usa-se o objeto amostra_fluido_aux para depois de
         setado ser incluido//
267    //no vetor de amostras
                                                                        //
268    //
         ////////////////////////////////////////////////////////////////////////
269
270    fin>>item_amostra;///<le a string "-chave"
271    fin>>amostra_fluido_aux.chave;///<le a chave da amostra i
272    fin>>item_amostra;///<le a string data
273    fin>>item_amostra;///<le a string  dia:
274    fin>>amostra_fluido_aux.dia;///<le a string
275    fin>>item_amostra;///<le a string  mes:
```

```
276     fin>>amostra_fluido_aux.mes;///<le a string
277     fin>>item_amostra;///<le a string  ano:
278     fin>>amostra_fluido_aux.ano;///<le a string
279     fin>>item_amostra;///<le a string -nome
280     fin>>amostra_fluido_aux.nome_do_fluido;///<le o nome do fluido
281     fin>>item_amostra;///<le a string -base
282     fin>>amostra_fluido_aux.base;///<le o nome da base
283     fin>>item_amostra;///<le a string -teorbase
284     fin>>amostra_fluido_aux.teorbase;///<le o teor da base
285     fin>>item_amostra;///<le a string -teor_de_agua
286     fin>>amostra_fluido_aux.teoragua;///<le o teor de agua
287     fin>>item_amostra;///<le a string linha
288     fin>>item_amostra;///<le a string PROPRIEDADES FÍSICAS E QUÍMICAS
289     fin>>item_amostra;///<le a string pH_min
290     fin>>amostra_fluido_aux.pH_min;///<le o pH_min
291     fin>>item_amostra;///<le a string pH_max
292     fin>>amostra_fluido_aux.pH_max;///<le o pH_max
293     fin>>item_amostra;///<le a string --pe_min
294     fin>>amostra_fluido_aux.pe_min;///<le pe_min
295     fin>>item_amostra;///<le a string --pe_max
296     fin>>amostra_fluido_aux.pe_max;///<le pe_max
297     fin>>item_amostra;///<le a string --temp_e
298     fin>>amostra_fluido_aux.temp_e;///<le temp_e
299     fin>>item_amostra;///<le a string --gel_i_ae
300     fin>>amostra_fluido_aux.gel_i_ae;///<le gel_i_ae
301     fin>>item_amostra;///<le a string --gel_f_ae
302     fin>>amostra_fluido_aux.gel_f_ae;///<le gel_f_ae
303     fin>>item_amostra;///<le a string --gel_i_de
304     fin>>amostra_fluido_aux.gel_i_de;///<le gel_i_de
305     fin>>item_amostra;///<le a string --gel_f_de
306     fin>>amostra_fluido_aux.gel_f_de;///<le gel_f_de
307     fin>>item_amostra;///<le a string --est_el_ae
308     fin>>amostra_fluido_aux.est_el_ae;///<le est_el_ae
309     fin>>item_amostra;///<le a string --est_el_de
310     fin>>amostra_fluido_aux.est_el_de;///<le est_el_de
311     fin>>item_amostra;///<le a string -c_lubricidade
312     fin>>amostra_fluido_aux.c_lubricidade;///<le c_lubricidade
313     fin>>item_amostra;///<le a string -filtrado
314     fin>>amostra_fluido_aux.filtrado;///<le filtrado
315     fin>>item_amostra;///<le a string -teor_solidos
316     fin>>amostra_fluido_aux.teor_solidos;///<le teor_solidos
317     fin>>item_amostra;///<le a string -salinidade
318     fin>>amostra_fluido_aux.salinidade;///<le salinidade
319     fin>>item_amostra;///<le a string linha
320     fin>>item_amostra;///<le a string PROPRIEDADES REOLÓGICAS
321     fin>>item_amostra;///<le a string VA_ae
322     fin>>amostra_fluido_aux.VA_ae;///<le VA_ae
323     fin>>item_amostra;///<le a string VA_de
```

```
324     fin>>amostra_fluido_aux.VA_de;///<le VA_de
325     fin>>item_amostra;///<le a string VP_ae
326     fin>>amostra_fluido_aux.VP_ae;///<le VP_ae
327     fin>>item_amostra;///<le a string VP_de
328     fin>>amostra_fluido_aux.VP_de;///<le VP_de
329     fin>>item_amostra;///<le a string LE_ae
330     fin>>amostra_fluido_aux.LE_ae;///<le LE_ae
331     fin>>item_amostra;///<le a string LE_de
332     fin>>amostra_fluido_aux.LE_de;///<le LE_de
333     fin>>item_amostra;///<le a string linha
334     fin>>item_amostra;///<le a string ADITIVOS
335     fin>>item_amostra;///<le a string adensante
336     fin>>amostra_fluido_aux.adensante;///<le adensante
337     fin>>item_amostra;///<le a string conc_adensante
338     fin>>amostra_fluido_aux.conc_adensante;///<le conc_adensante
339     fin>>item_amostra;///<le a string inibidor_fa
340     fin>>amostra_fluido_aux.inibidor_fa;///<le inibidor_fa
341     fin>>item_amostra;///<le a string conc_inibidor_fa
342     fin>>amostra_fluido_aux.conc_inibidor_fa;///<le conc_inibidor_fa
343     fin>>item_amostra;///<le a string redutor_f
344     fin>>amostra_fluido_aux.redutor_f;///<le redutor_f
345     fin>>item_amostra;///<le a string conc_redutor_f
346     fin>>amostra_fluido_aux.conc_redutor_f;///<le conc_redutor_f
347     fin>>item_amostra;///<le a string biopolimero
348     fin>>amostra_fluido_aux.biopolimero;///<le biopolimero
349     fin>>item_amostra;///<le a string conc_biopolimero
350     fin>>amostra_fluido_aux.conc_biopolimero;///<le conc_biopolimero
351     fin>>item_amostra;///<le a string viscosificante
352     fin>>amostra_fluido_aux.viscosificante;///<le viscosificante
353     fin>>item_amostra;///<le a string conc_viscosificante
354     fin>>amostra_fluido_aux.conc_viscosificante;///<le
          conc_viscosificante
355     fin>>item_amostra;///<le a string dispersante
356     fin>>amostra_fluido_aux.dispersante;///<le dispersante
357     fin>>item_amostra;///<le a string conc_dispersante
358     fin>>amostra_fluido_aux.conc_dispersante;///<le conc_dispersante
359     fin>>item_amostra;///<le a string defloculante
360     fin>>amostra_fluido_aux.defloculante;///<le defloculante
361     fin>>item_amostra;///<le a string conc_defloculante
362     fin>>amostra_fluido_aux.conc_defloculante;///<le conc_defloculante
363     fin>>item_amostra;///<le a string emulsificante
364     fin>>amostra_fluido_aux.emulsificante;///<le emulsificante
365     fin>>item_amostra;///<le a string conc_emulsificante
366     fin>>amostra_fluido_aux.conc_emulsificante;///<le conc_emulsificante
367     fin>>item_amostra;///<le a string biocida
368     fin>>amostra_fluido_aux.biocida;///<le biocida
369     fin>>item_amostra;///<le a string conc_biocida
370     fin>>amostra_fluido_aux.conc_biocida;///<le conc_biocida
```

```
371     fin>>item_amostra;///<le a string lubrificante
372     fin>>amostra_fluido_aux.lubrificante;///<le lubrificante
373     fin>>item_amostra;///<le a string conc_lubrificante
374     fin>>amostra_fluido_aux.conc_lubrificante;///<le conc_lubrificante
375     fin>>item_amostra;///<le a string inibidor_c
376     fin>>amostra_fluido_aux.inibidor_c;///<le inibidor_c
377     fin>>item_amostra;///<le a string conc_inibidor_c
378     fin>>amostra_fluido_aux.conc_inibidor_c;///<le conc_inibidor_c
379     fin>>item_amostra;///<le a string controlador_pH
380     fin>>amostra_fluido_aux.controlador_pH;///<le controlador_pH
381     fin>>item_amostra;///<le a string controlador_pH
382     fin>>amostra_fluido_aux.conc_controlador_pH;///<le
            conc_controlador_pH
383
384     amostra_fluido.push_back(amostra_fluido_aux);
385     fin.close();
386
387   }
388   return amostra_fluido;
389
390 }
391
392 vector<CFluidodePerfuracao> CBancodeDados::ler_amostra_basico(int
       filtro_)///<funcao listar amostras
393 {
394   vector<CFluidodePerfuracao> amostra_fluido_vec_aux;
395   amostra_fluido_vec_aux.clear();
396   amostra_fluido_vec_aux=ler_amostra_basico();///<le tododas infos
        basicas
397   ///<de todas amostras
398   vector<CFluidodePerfuracao> amostra_fluido;
399   amostra_fluido.clear();
400   int filtro;
401   filtro=filtro_;
402   int int_chave;
403   unsigned int c;
404   double dado_double;
405   string nome;
406   ///<comeca preencher o vetor de amostras amostra_fluifo
407   amostra_fluido.push_back(amostra_fluido_vec_aux[0]);///<retorna a
        amostra zero como base de calculos
408   switch (filtro)///<testa a resposta
409
410   {
411     case 1:
412       cout<<"Informe␣a␣CHAVE:␣";
413       cin>>int_chave;
414       cin.get();
```

```
415       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
416       {
417         if(amostra_fluido_vec_aux[c].chave==int_chave)
418         {
419           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
420         }
421       }
422
423       break;
424
425     case 2:
426       cout<<"Informe o nome do Fluido: ";
427       cin>>nome;
428       cin.get();
429       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
430       {
431         if(amostra_fluido_vec_aux[c].nome_do_fluido==nome)
432         {
433           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
434         }
435       }
436
437       break;
438
439     case 3:
440       cout<<"Informe a Base: ";
441       cin>>nome;
442       cin.get();
443       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
444       {
445         if(amostra_fluido_vec_aux[c].base==nome)
446         {
447           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
448         }
449       }
450
451       break;
452
453     case 4:
454       cout<<"Informe o pH minimo: ";
455
456
457
458       cin>>nome;
459       cin.get();
460
461       if(nome=="-")
462       {
```

```
463         dado_double=999999;
464       }
465       else
466       {
467         if(atof(nome.c_str()))
468         {
469           dado_double=atof(nome.c_str());
470         }
471         else
472         {
473           dado_double=999999;
474         }
475       }
476 
477 
478       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
479       {
480         if(amostra_fluido_vec_aux[c].pH_min==dado_double)
481         {
482           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
483         }
484       }
485 
486       break;
487 
488     case 5:
489       cout<<"Informe␣o␣pH␣maximo:␣";
490       cin>>nome;
491       cin.get();
492 
493       if(nome=="-")
494       {
495         dado_double=999999;
496       }
497       else
498       {
499         if(atof(nome.c_str()))
500         {
501           dado_double=atof(nome.c_str());
502         }
503         else
504         {
505           dado_double=999999;
506         }
507       }
508 
509       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
510       {
```

```
511         if(amostra_fluido_vec_aux[c].pH_max==dado_double)
512         {
513           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
514         }
515       }
516 
517       break;
518 
519 
520 
521     case 6:
522       cout<<"Informe␣o␣Teor␣de␣agua:␣";
523       cin>>nome;
524       cin.get();
525 
526       if(nome=="-")
527       {
528         dado_double=999999;
529       }
530       else
531       {
532         if(atof(nome.c_str()))
533         {
534           dado_double=atof(nome.c_str());
535         }
536         else
537         {
538           dado_double=999999;
539         }
540       };
541       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
542       {
543         if(amostra_fluido_vec_aux[c].teoragua==dado_double)
544         {
545           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
546         }
547       }
548 
549       break;
550 
551     case 7:
552       cout<<"Informe␣o␣peso␣especifico␣minimo:␣";
553        cin>>nome;
554       cin.get();
555 
556 
557       if(nome=="-")
558       {
```

```
559         dado_double=999999;
560       }
561       else
562       {
563         if(atof(nome.c_str()))
564         {
565           dado_double=atof(nome.c_str());
566         }
567         else
568         {
569           dado_double=999999;
570         }
571       }
572 
573 
574       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
575       {
576         if(amostra_fluido_vec_aux[c].pe_min==dado_double)
577         {
578           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
579         }
580       }
581 
582       break;
583 
584 
585 
586     case 8:
587       cout<<"Informe␣o␣peso␣especifico␣maximo:␣";
588       cin>>nome;
589       cin.get();
590 
591       if(nome=="-")
592       {
593         dado_double=999999;
594       }
595       else
596       {
597         if(atof(nome.c_str()))
598         {
599           dado_double=atof(nome.c_str());
600         }
601         else
602         {
603           dado_double=999999;
604         }
605       }
606 
```

```
607
608       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
609       {
610         if(amostra_fluido_vec_aux[c].pe_max==dado_double)
611         {
612           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
613         }
614       }
615
616       break;
617
618
619
620
621     case 9:
622       cout<<"Informe␣a␣Temperatura␣de␣envelhecimento:␣";
623       cin>>nome;
624       cin.get();
625
626       if(nome=="-")
627       {
628         dado_double=999999;
629       }
630       else
631       {
632         if(atof(nome.c_str()))
633         {
634           dado_double=atof(nome.c_str());
635         }
636         else
637         {
638           dado_double=999999;
639         }
640       }
641       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
642       {
643         if(amostra_fluido_vec_aux[c].temp_e==dado_double)
644         {
645           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
646         }
647       }
648
649       break;
650
651     case 10:
652       cout<<"Informe␣a␣Salinidade:␣";
653       cin>>nome;
654       cin.get();
```

```
655
656       if(nome=="-")
657       {
658         dado_double=999999;
659       }
660       else
661       {
662         if(atof(nome.c_str()))
663         {
664           dado_double=atof(nome.c_str());
665         }
666         else
667         {
668           dado_double=999999;
669         }
670       }
671       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
672       {
673         if(amostra_fluido_vec_aux[c].salinidade==dado_double)
674         {
675           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
676         }
677       }
678
679       break;
680
681     case 11:
682       cout<<"Informe␣o␣volume␣de␣filtrado:␣";
683       cin>>nome;
684       cin.get();
685
686       if(nome=="-")
687       {
688         dado_double=999999;
689       }
690       else
691       {
692         if(atof(nome.c_str()))
693         {
694           dado_double=atof(nome.c_str());
695         }
696         else
697         {
698           dado_double=999999;
699         }
700       }
701       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
702       {
```

```
703         if(amostra_fluido_vec_aux[c].filtrado==dado_double)
704         {
705           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
706         }
707       }
708 
709       break;
710 
711     case 12:
712       cout<<"Informe o Dia: ";
713       cin>>dia;
714       cin.get();
715       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
716       {
717         if(amostra_fluido_vec_aux[c].dia==dia)
718         {
719           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
720         }
721       }
722 
723       break;
724 
725     case 13:
726       cout<<"Informe o Mes: ";
727       cin>>mes;
728       cin.get();
729       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
730       {
731         if(amostra_fluido_vec_aux[c].mes==mes)
732         {
733           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
734         }
735       }
736 
737       break;
738 
739 
740     case 14:
741       cout<<"Informe o ano: ";
742       cin>>ano;
743       cin.get();
744       for(c=0;c<amostra_fluido_vec_aux.size();c++)///<corre amostras
745       {
746         if(amostra_fluido_vec_aux[c].ano==ano)
747         {
748           amostra_fluido.push_back(amostra_fluido_vec_aux[c]);
749         }
750       }
```

```
751
752       break;
753
754     case 15: ///< listar todos
755       amostra_fluido=amostra_fluido_vec_aux;
756       break;
757
758     default:
759       cout<<"Filtro invalido."<<endl;
760       ///<verifica sistema operacional
761       #ifdef __linux__
762       ///<sleep(1000);
763       #elif _WIN32
764       Sleep(1000);
765       #else
766
767       #endif
768       resposta=1;
769       break;
770   }
771
772
773   return amostra_fluido;
774
775}
776
777void CBancodeDados::exibir_amostra(int chave_)
778{
779
780   int_chave=chave_;
781   CFluidodePerfuracao amostra_aux;
782   vector<CFluidodePerfuracao> v_amostra;
783   unsigned int c;
784   ///lendo todas as amostras basico
785   v_amostra=ler_amostra_basico();
786   ///procurando amostra correspontende a chave informada
787   for(c=0;c<v_amostra.size();c++)
788   {
789     if (int_chave==v_amostra[c].chave)
790     {string teste_vazio(double);
791       double teste_vazio(string);
792       amostra_aux=v_amostra[c];
793       break;
794     }
795   }
796   ///inicia das exibicoes basica das amostras
797   cout<<"------------------------------------------------"<<endl;
798   cout<<"                                                "<<endl;
```

```
799 cout<<"␣␣␣␣␣␣Exibindo␣Amostra␣de␣Chave␣"<<int_chave<<"␣␣␣␣␣"<<endl;
800 cout<<"_____________________________________________"<<endl;
801 cout<<endl<<endl;
802 cout<<"Data␣de␣Cadastro␣----␣:␣"<<amostra_aux.dia<<"/"<<amostra_aux.
       mes<<"/"<<amostra_aux.ano<<endl;
803 cout<<"Nome␣do␣Fluido␣-------:␣"<<amostra_aux.nome_do_fluido<<endl;
804 cout<<"Base␣----------------␣:␣"<<amostra_aux.base<<endl;
805 cout<<"Teor␣de␣Base␣--------␣:␣"<<teste_vazio(amostra_aux.teorbase)<<
       endl;
806 cout<<"Teor␣de␣Agua␣--------␣:␣"<<teste_vazio(amostra_aux.teoragua)<<
       endl<<endl;
807 cout<<"␣␣␣␣-----Propriedades␣Fisicas␣e␣Quimicas-----␣␣␣␣␣␣"<<endl<<
       endl;
808 cout<<"pH␣Minimo␣-----------␣:␣"<<teste_vazio(amostra_aux.pH_min)<<
       endl;
809 cout<<"pH␣Maximo␣-----------␣:␣"<<teste_vazio(amostra_aux.pH_max)<<
       endl;
810 cout<<"Peso␣Especifico␣Minimo:␣"<<teste_vazio(amostra_aux.pe_min)<<
       endl;
811 cout<<"Peso␣Especifico␣Maximo:␣"<<teste_vazio(amostra_aux.pe_max)<<
       endl;
812 cout<<"Temperatura␣de␣envelhecimento:␣"<<teste_vazio(amostra_aux.
       temp_e)<<endl;
813 cout<<"Forca␣gel␣inicial␣antes␣do␣envelhecimento:␣"<<teste_vazio(
       amostra_aux.gel_i_ae)<<endl;
814 cout<<"Forca␣gel␣final␣antes␣do␣envelhecimento:␣"<<teste_vazio(
       amostra_aux.gel_f_ae)<<endl;
815 cout<<"Forca␣gel␣inicial␣depois␣do␣envelhecimento:␣"<<teste_vazio(
       amostra_aux.gel_i_de)<<endl;
816 cout<<"Forca␣gel␣final␣depois␣do␣envelhecimento:␣"<<teste_vazio(
       amostra_aux.gel_f_de)<<endl;
817 cout<<"Estabilidade␣eletrica␣antes␣do␣envelhecimento:␣"<<teste_vazio(
       amostra_aux.est_el_ae)<<endl;
818 cout<<"Estabilidade␣eletrica␣depois␣do␣envelhecimento:␣"<<teste_vazio(
       amostra_aux.est_el_de)<<endl;
819 cout<<"Coeficiente␣de␣lubricidade:␣"<<teste_vazio(amostra_aux.
       c_lubricidade)<<endl;
820 cout<<"Volume␣de␣filtrado:␣"<<teste_vazio(amostra_aux.filtrado)<<endl;
821 cout<<"Teor␣de␣solidos␣----------␣:␣"<<teste_vazio(amostra_aux.
       teor_solidos)<<endl;
822 cout<<"Salinidade␣----------␣:␣"<<teste_vazio(amostra_aux.salinidade)
       <<endl<<endl;
823 cout<<"␣␣␣␣----Propriedades␣Reologicas----␣␣␣␣␣␣"<<endl<<endl;
824 cout<<"Viscosidade␣Aparente␣antes␣do␣envelhecimento:␣"<<teste_vazio(
       amostra_aux.VA_ae)<<endl;
825 cout<<"Viscosidade␣Aparente␣depois␣do␣envelhecimento:␣"<<teste_vazio(
       amostra_aux.VA_de)<<endl;
826 cout<<"Viscosidade␣Plastica␣antes␣do␣envelhecimento:␣"<<teste_vazio(
```

```
        amostra_aux.VP_ae)<<endl;
827   cout<<"Viscosidade␣Plastica␣depois␣do␣envelhecimento:␣"<<teste_vazio(
        amostra_aux.VP_de)<<endl<<endl;
828   cout<<"Limite␣de␣escoamento␣antes␣do␣envelhecimento:␣"<<teste_vazio(
        amostra_aux.LE_ae)<<endl<<endl;
829   cout<<"Limite␣de␣escoamento␣depois␣do␣envelhecimento:␣"<<teste_vazio(
        amostra_aux.LE_de)<<endl<<endl;
830   cout<<"␣␣␣␣----Aditivos----␣␣␣␣␣␣"<<endl<<endl;
831   cout<<"Adensante␣----------␣:␣"<<amostra_aux.adensante<<endl;
832   cout<<"Concentracao␣de␣Adensante␣----------␣:␣"<<teste_vazio(
        amostra_aux.conc_adensante)<<endl;
833   cout<<"Inibidor␣de␣Formacoes␣Ativas␣-------␣:␣"<<amostra_aux.
        inibidor_fa<<endl;
834   cout<<"Concentracao␣do␣Inibidor␣de␣Formacoes␣Ativas␣----------␣:␣"<<
        teste_vazio(amostra_aux.conc_inibidor_fa)<<endl;
835   cout<<"Redutor␣de␣Filtrado␣␣:␣"<<amostra_aux.redutor_f<<endl;
836   cout<<"Concentracao␣do␣Redutor␣de␣Filtrado␣----------␣:␣"<<teste_vazio
        (amostra_aux.conc_redutor_f)<<endl;
837   cout<<"Biopolimero␣␣:␣"<<amostra_aux.biopolimero<<endl;
838   cout<<"Concentracao␣do␣Biopolimero␣----------␣:␣"<<teste_vazio(
        amostra_aux.conc_biopolimero)<<endl;
839   cout<<"Viscosificante␣␣:␣"<<amostra_aux.viscosificante<<endl;
840   cout<<"Concentracao␣do␣Viscosificante----------␣:␣"<<teste_vazio(
        amostra_aux.conc_viscosificante)<<endl;
841   cout<<"Dispersante␣␣:␣"<<amostra_aux.dispersante<<endl;
842   cout<<"Concentracao␣do␣Dispersante----------␣:␣"<<teste_vazio(
        amostra_aux.conc_dispersante)<<endl;
843   cout<<"Defloculante␣␣:␣"<<amostra_aux.defloculante<<endl;
844   cout<<"Concentracao␣do␣Defloculante----------␣:␣"<<teste_vazio(
        amostra_aux.conc_defloculante)<<endl;
845   cout<<"Emulsificante␣␣:␣"<<amostra_aux.emulsificante<<endl;
846   cout<<"Concentracao␣do␣Emulsificante----------␣:␣"<<teste_vazio(
        amostra_aux.conc_emulsificante)<<endl;
847   cout<<"Biocida␣␣:␣"<<amostra_aux.biocida<<endl;
848   cout<<"Concentracao␣do␣Biocida----------␣:␣"<<teste_vazio(amostra_aux.
        conc_biocida)<<endl;
849   cout<<"Lubrificante␣␣:␣"<<amostra_aux.lubrificante<<endl;
850   cout<<"Concentracao␣do␣Lubrificante----------␣:␣"<<teste_vazio(
        amostra_aux.conc_lubrificante)<<endl;
851   cout<<"Inibidor␣de␣corrosao␣␣:␣"<<amostra_aux.inibidor_c<<endl;
852   cout<<"Concentracao␣do␣Inibidor␣de␣Corrosao----------␣:␣"<<teste_vazio
        (amostra_aux.conc_inibidor_c)<<endl;
853   cout<<"Controlador␣de␣pH␣␣:␣"<<amostra_aux.controlador_pH<<endl;
854   cout<<"Concentracao␣do␣Controlador␣de␣pH----------␣:␣"<<teste_vazio(
        amostra_aux.conc_controlador_pH)<<endl;
855   cout<<endl<<endl;
856 }
857
```

```
858 void CBancodeDados::listar_amostra(vector<CFluidodePerfuracao>
      amostra_fluido_)///<funcao listar amostras
859 {
860
861   i=0;///<inteiro para contagem no for
862
863   amostra_fluido=amostra_fluido_;
864   ////////////////////////////////////////////
865   //Inicia a listagem das amostras lidas no disco
866   ////////////////////////////////////////////
867   //escreve linha superior ao titulo dos atributos das amostras
868   for(i=0;i<90;i++)
869   {
870
871     #ifdef __linux
872     cout<<"\u2500";
873     #elif _WIN32
874     cout<<(char)196;
875     #else
876
877     #endif
878
879   }
880   cout<<endl;//finaliza linha
881   ///<escreve titulo dos atributos das amostras
882   cout.setf(ios::left);
883   cout.width(7);
884   cout<<"CHAVE";
885
886   cout.setf(ios::left);
887   cout.width(20);
888   cout<<"NOME␣DO␣FLUIDO";
889
890   cout.setf(ios::left);
891   cout.width(10);
892   cout<<"BASE";
893
894   cout.setf(ios::left);
895   cout.width(20);
896   cout<<"TEOR␣DE␣BASE";
897
898   cout.setf(ios::left);
899   cout.width(20);
900   cout<<"TEOR␣DE␣AGUA";
901
902   cout.setf(ios::left);
903   cout.width(20);
904   cout<<"DATA";
```

```
905
906   cout<<endl;///finaliza titulo dos itens
907
908   ///escreve a linha abaixo do titulo das tabelas
909   for(i=0;i<90;i++)
910   {
911     #ifdef __linux
912     cout<<"\u2500";
913     #elif _WIN32
914     cout<<(char)196;
915     #else
916
917     #endif
918   }
919
920   ///escreve os atributos das amostras na tela
921
922   cout<<endl;///<pula linha para primeira listagem
923   for(i=0;i<amostra_fluido.size();i++)///</<comeca listar da 1, amostra
          zero é a base
924   {
925
926     if(amostra_fluido[i].chave!=0)
927     {
928
929       cout.setf(ios::left);
930       cout.width(7);
931       cout<<amostra_fluido[i].chave;///<escreve as chaves
932
933       cout.setf(ios::left);
934       cout.width(20);
935       cout<<amostra_fluido[i].nome_do_fluido;///<escreve o nome do
              fluido
936
937       cout.setf(ios::left);
938       cout.width(10);
939       cout<<amostra_fluido[i].base;///<escreve a base
940
941       cout.setf(ios::left);
942       cout.width(20);
943       cout<<teste_vazio(amostra_fluido[i].teorbase);///<escreve o teor
              de base
944
945       cout.setf(ios::left);
946       cout.width(20);
947       cout<<teste_vazio(amostra_fluido[i].teoragua);///<escreve o teor
              de agua
948
```

```
949       cout.setf(ios::left);
950       cout.width(20);
951       if(amostra_fluido[i].dia<10)ss<<0;
952                           ss<<amostra_fluido[i].dia<<"/";
953       if(amostra_fluido[i].mes<10)ss<<0;
954                           ss<<amostra_fluido[i].mes<<"/";
955       if(amostra_fluido[i].ano<10)ss<<0;
956                           ss<<amostra_fluido[i].ano;///<constroi data
957                           cout<<ss.str();
958       ss.str("");
959
960       cout<<endl;///<pula linha antes das pausas
961
962     }
963   }
964
965   i=0;
966   cout<<endl;
967
968   ///escreve linha inferior
969   for(i=0;i<90;i++)
970   {
971     #ifdef __linux
972     cout<<"\u2500";
973     #elif _WIN32
974     cout<<(char)196;
975     #else
976
977     #endif
978   }
979   cout<<endl;
980   chave.clear();
981 }
982
983 void CBancodeDados::listar_amostra(vector<CFluidodePerfuracao>
      amostra_fluido_, int filtro_)//funcao listar amostras
984 {
985
986   i=0;
987   filtro=filtro_;
988   amostra_fluido=amostra_fluido_;
989   ////////////////////////////////////////////
990   //Inicia a listagem das amostras lidas no disco
991   ////////////////////////////////////////////
992   //escreve linha superior ao titulo dos atributos das amostras
993   for(i=0;i<80;i++)
994   {
995
```

```
996     #ifdef __linux
997     cout<<"\u2500";
998     #elif _WIN32
999     cout<<(char)196;
1000    #else
1001
1002    #endif
1003
1004  }
1005  cout<<endl;//finaliza linha
1006  ///<escreve titulo dos atributos das amostras
1007  cout.setf(ios::left);
1008  cout.width(10);
1009  cout<<"CHAVE";
1010
1011  cout.setf(ios::left);
1012  cout.width(20);
1013  cout<<"NOME␣DO␣FLUIDO";
1014
1015  cout.setf(ios::left);
1016  cout.width(20);
1017  cout<<"TEOR␣DE␣BASE";
1018
1019  switch(filtro)
1020  {
1021
1022    case 3:
1023      cout.setf(ios::left);
1024      cout.width(20);
1025      cout<<"Base";
1026      break;
1027
1028    case 4:
1029      cout.setf(ios::left);
1030      cout.width(20);
1031      cout<<"pH␣minimo";
1032      break;
1033
1034    case 5:
1035      cout.setf(ios::left);
1036      cout.width(20);
1037      cout<<"pH␣maximo";
1038      break;
1039
1040    case 6:
1041      cout.setf(ios::left);
1042      cout.width(20);
1043      cout<<"Teor␣de␣agua";
```

```
1044        break;
1045
1046      case 7:
1047        cout.setf(ios::left);
1048        cout.width(20);
1049        cout<<"Peso␣especifico␣minimo";
1050        break;
1051
1052      case 8:
1053        cout.setf(ios::left);
1054        cout.width(20);
1055        cout<<"Peso␣especifico␣maximo";
1056        break;
1057
1058      case 9:
1059        cout.setf(ios::left);
1060        cout.width(20);
1061        cout<<"Temperatura␣de␣envelhecimento";
1062        break;
1063
1064      case 10:
1065        cout.setf(ios::left);
1066        cout.width(20);
1067        cout<<"Salinidade";
1068        break;
1069
1070      case 11:
1071        cout.setf(ios::left);
1072        cout.width(20);
1073        cout<<"Filtrado";
1074        break;
1075
1076      case 12:
1077        cout.setf(ios::left);
1078        cout.width(10);
1079        cout<<"Dia";
1080        break;
1081
1082       case 13:
1083        cout.setf(ios::left);
1084        cout.width(10);
1085        cout<<"Mes";
1086        break;
1087
1088      case 14:
1089        cout.setf(ios::left);
1090        cout.width(10);
1091        cout<<"Ano";
```

```
1092        break;
1093
1094    }
1095    cout<<endl;//finaliza titulo dos itens
1096
1097    ///escreve a linha abaixo do titulo das tabelas
1098    for(i=0;i<80;i++)
1099    {
1100      #ifdef __linux
1101      cout<<"\u2500";
1102      #elif _WIN32
1103      cout<<(char)196;
1104      #else
1105
1106      #endif
1107    }
1108
1109    ///<escreve os atributos das amostras na tela
1110    for(i=0;i<amostra_fluido.size();i++)
1111    {
1112      if(amostra_fluido[i].chave!=0) ///<nao lista amostra de chave zero
1113      {
1114
1115
1116        cout<<endl;
1117
1118        cout.setf(ios::left);
1119        cout.width(10);
1120        cout<<amostra_fluido[i].chave;///<escreve as chaves
1121
1122        cout.setf(ios::left);
1123        cout.width(20);
1124        cout<<amostra_fluido[i].nome_do_fluido;///<escreve o nome do
              fluido
1125
1126        cout.setf(ios::left);
1127        cout.width(20);
1128        cout<<teste_vazio(amostra_fluido[i].teorbase);///<escreve o teor
              de base
1129
1130
1131        switch(filtro)
1132        {
1133
1134          case 3:
1135            cout.setf(ios::left);
1136            cout.width(20);
1137            cout<<teste_vazio(amostra_fluido[i].base);
```

```
1138            break;
1139
1140          case 4:
1141            cout.setf(ios::left);
1142            cout.width(20);
1143            cout<<teste_vazio(amostra_fluido[i].pH_min);
1144            break;
1145
1146          case 5:
1147            cout.setf(ios::left);
1148            cout.width(20);
1149            cout<<teste_vazio(amostra_fluido[i].pH_max);
1150            break;
1151
1152          case 6:
1153            cout.setf(ios::left);
1154            cout.width(20);
1155            cout<<teste_vazio(amostra_fluido[i].teoragua);
1156            break;
1157
1158          case 7:
1159            cout.setf(ios::left);
1160            cout.width(20);
1161            cout<<teste_vazio(amostra_fluido[i].pe_min);
1162            break;
1163
1164          case 8:
1165            cout.setf(ios::left);
1166            cout.width(20);
1167            cout<<teste_vazio(amostra_fluido[i].pe_max);
1168            break;
1169
1170          case 9:
1171            cout.setf(ios::left);
1172            cout.width(20);
1173            cout<<teste_vazio(amostra_fluido[i].temp_e);
1174            break;
1175
1176          case 10:
1177            cout.setf(ios::left);
1178            cout.width(20);
1179            cout<<teste_vazio(amostra_fluido[i].salinidade);
1180            break;
1181
1182          case 11:
1183            cout.setf(ios::left);
1184            cout.width(20);
1185            cout<<teste_vazio(amostra_fluido[i].filtrado);
```

```
1186            break;
1187
1188          case 12:
1189            cout.setf(ios::left);
1190            cout.width(20);
1191            if(amostra_fluido[i].dia<10)ss<<0;
1192                          ss<<amostra_fluido[i].dia;
1193            cout<<ss.str();
1194            ss.str("");
1195            break;
1196
1197          case 13:
1198            cout.setf(ios::left);
1199            cout.width(20);
1200            if(amostra_fluido[i].mes<10)ss<<0;
1201                          ss<<amostra_fluido[i].mes;
1202            cout<<ss.str();
1203            ss.str("");
1204            break;
1205
1206          case 14:
1207            cout.setf(ios::left);
1208            cout.width(20);
1209            if(amostra_fluido[i].ano<10)ss<<0;
1210                          ss<<amostra_fluido[i].ano;
1211            cout<<ss.str();
1212            ss.str("");
1213            break;
1214        }
1215
1216      }
1217
1218
1219      }
1220      i=0;
1221      cout<<endl;
1222
1223      ///escreve linha inferior
1224      for(i=0;i<80;i++)
1225      {
1226        #ifdef __linux
1227        cout<<"\u2500";
1228        #elif _WIN32
1229        cout<<(char)196;
1230        #else
1231
1232        #endif
1233      }
```

```
1234    cout<<endl;
1235    chave.clear();
1236  }
1237
1238  void CBancodeDados::excluir_amostra(int int_chave_)
1239  {
1240    ////////////////////////////////////
1241    //Inicio da exclusao chave
1242    ////////////////////////////////////
1243    vector<int> chave;///<vetor que armazena as chaves
1244
1245    int_chave=int_chave_;///<inteiro que recebe chave informada
1246
1247    //abre o arquivo de informacoes do bd
1248    #ifdef __linux
1249    caminho_bd<<"bd"<<char(47)<<"ini"<<char(47)<<"bdinfo";
1250    #elif _WIN32
1251    caminho_bd<<"bd"<<char(92)<<"ini"<<char(92)<<"bdinfo";
1252    #else
1253
1254    #endif
1255    fin.open(caminho_bd.str().c_str());
1256    caminho_bd.str("");
1257    ///<carrega o arquivo de informacoes do bd
1258    while(! fin.eof())
1259    {
1260      fin >> int_chave_aux;
1261      chave.push_back(int_chave_aux);
1262    }
1263    fin.close();
1264    ///<inicia pesquisa da chave a ser deletada
1265    for(i=0;i<=chave.size();i++)
1266    {
1267      if (chave[i]==int_chave)
1268      {
1269        chave.erase(chave.begin() + i);///<deleta a chave encontrada
1270      }
1271    }
1272
1273    ///reescreve vetor no arquivo bdinfo
1274    ofstream fout;
1275    #ifdef __linux
1276    caminho_bd<<"bd"<<char(47)<<"ini"<<char(47)<<"bdinfo";
1277    #elif _WIN32
1278    caminho_bd<<"bd"<<char(92)<<"ini"<<char(92)<<"bdinfo";
1279    #else
1280
1281    #endif
```

```
1282    fout.open(caminho_bd.str().c_str());///<abre bdinfo para escrever
            informacoes atualizadas
1283    caminho_bd.str("");
1284    ///testa se nao abriu o arquivo
1285    if(fout.fail())
1286    {
1287      cout<<"Nao␣foi␣possivel␣reescrever␣a␣chave."<<endl;
1288    }
1289    for(i=0;i<(chave.size()-1);i++)///<escreve ate o punultimo item do
            vetor chave pulando linha
1290    {
1291      fout<<chave[i]<<endl;
1292    }
1293    ///<para nÃ£o pular linha ao fim do arquivo, escreve o ultimo item
            do vetor chave
1294    fout<<chave[i];
1295    fout.close();///<fecha o arquivo de informacoes do banco de dados
1296    i=0;
1297    ////////////////////////////////////
1298    //Fim da exclusao chave
1299    ////////////////////////////////////
1300    ////////////////////////////////////
1301    //Inicio da remocao da amostra para \lixo
1302    ////////////////////////////////////
1303    stringstream comando_remover;///<DeclaraÃ§ao das variaveis do
            comando de remover para lixeira
1304    ///criar o comando a ser executado
1305    ///comando: "move bd\0.txt bd\lixo" Comando para mover o arquivo da
            amostra
1306    comando_remover.str("");///<apagand stringstream
1307    ///verifica sistema operacional
1308    #ifdef __linux
1309    ///mv ./bd/0.txt ./bd/lixo
1310    comando_remover<<"mv"<<"␣"<<"./bd/"<<int_chave<<"␣"<<"./bd/lixo";
1311    #elif _WIN32
1312    comando_remover<<"move"<<"␣"<<"bd"<<char(92)<<int_chave<<"␣"<<"bd"<<
            char(92)<<"lixo";
1313
1314    #else
1315
1316    #endif
1317    system(comando_remover.str().c_str());///<Manda o comando pro prompt
             para mover o arquivo p lixeira
1318    comando_remover.str("");///<apagand stringstream
1319
1320    }
1321
1322  string CBancodeDados::teste_vazio(double num_double_)
```

```
1323  {
1324    num_double_ss.str("");
1325    num_double=num_double_;
1326    const_vazio_num_double=999999;
1327    if(num_double==const_vazio_num_double)
1328    {
1329      return("-");
1330    }
1331    else
1332    {
1333      num_double_ss<<num_double;///<converte o double para string
1334      return(num_double_ss.str());
1335    }
1336  }
1337
1338  double CBancodeDados::teste_vazio(string vazio_str_)
1339  {
1340    double dado_double;
1341    vazio_str=vazio_str_;
1342    const_vazio_str="-";
1343    if(vazio_str==const_vazio_str)
1344    {
1345      return 999999;
1346    }
1347    else
1348    {
1349      return dado_double=atof(vazio_str.c_str());///<converte a string
           que entrou para double
1350    }
1351  }
1352
1353  void CBancodeDados::inserir_amostra()
1354  {
1355    ///Inclusao de chave no arquivo bdinfo
1356
1357    vector<int> chave;///<vetor que armazena as chaves
1358    ifstream fin;///<objeto de leitura
1359    int int_chave;///<inteiro que recebe cada linha lida no arquivo
1360
1361    #ifdef __linux
1362    caminho_bd<<"bd"<<char(47)<<"ini"<<char(47)<<"bdinfo";
1363    #elif _WIN32
1364    caminho_bd<<"bd"<<char(92)<<"ini"<<char(92)<<"bdinfo";
1365    #else
1366
1367    #endif
1368
1369    fin.open(caminho_bd.str().c_str());///<abre o arquivo de informacoes
```

```
        do bd usando metodos que retorna uma string de um numero e
       converte a string C++ para para padrao C
1370    caminho_bd.str("");
1371    while(! fin.eof())///<carrega o arquivo de informacoes do bd(chaves)
            e realiza leitura até o final do arquivo
1372    {
1373      fin >> int_chave;
1374      chave.push_back(int_chave);
1375    }
1376    fin.close();
1377
1378    ///Escrita da nova chave
1379    #ifdef __linux
1380    caminho_bd<<"bd"<<char(47)<<"ini"<<char(47)<<"bdinfo";
1381    #elif _WIN32
1382    caminho_bd<<"bd"<<char(92)<<"ini"<<char(92)<<"bdinfo";
1383    #else
1384
1385    #endif
1386
1387    ofstream fout;
1388    fout.open(caminho_bd.str().c_str(),ios::app);///<abre bdinfo e vai
           para o final do arquivo
1389    caminho_bd.str("");
1390    ///testa se nao abriu o arquivo
1391    if(fout.fail())
1392    {
1393      cout<<"Nao␣foi␣possivel␣inserir␣amostra."<<endl;
1394    }
1395
1396    int_chave++;///<soma ultima chave mais um
1397    fout<<endl;///<pula uma linha para inserir a chave
1398    fout<<int_chave;///<escreve a chave posterior no fim do arquivo
1399
1400    fout.close();
1401
1402    ///Fim da inclusao da chave
1403
1404    ///Inclusao do novo arquivo para novo fluido
1405
1406    ///cria diretorio
1407    #ifdef __linux
1408    comando_caminho_amostra<<"mkdir␣"<<"bd"<<char(47)<<int_chave;///<
           comando mkdir cria diretorio
1409    #elif _WIN32
1410    comando_caminho_amostra<<"mkdir␣"<<"bd"<<char(92)<<int_chave;
1411    #else
1412
```

```
1413    #endif
1414
1415    system(comando_caminho_amostra.str().c_str());///<cria o diretorio a
            ser gravado os arquivos
1416    comando_caminho_amostra.str("");///<apaga stringstream
1417
1418    ///<Formar nome do arquivo
1419
1420    #ifdef __linux
1421    nome_amostra<<"bd"<<char(47)<<int_chave<<char(47)<<int_chave;
1422    #elif _WIN32
1423    nome_amostra<<"bd"<<char(92)<<int_chave<<char(92)<<int_chave;
1424    #else
1425
1426    #endif
1427
1428    ///Escrita no arquivo do novo fluido
1429
1430    fout.open(nome_amostra.str().c_str());///<abre o arquivo
1431    nome_amostra.str("");
1432
1433    string nome;
1434    float num;
1435
1436    cout<<"---------------------------------"<<endl;
1437    cout<<"Preencha␣com␣os␣dados␣da␣amostra:␣"<<endl;
1438    cout<<"Para␣dado␣numerico␣nulo,␣informe␣"<<endl;
1439    cout<<"apenas␣um␣-␣␣␣␣␣␣␣␣.␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"<<endl;
1440    cout<<"---------------------------------"<<endl;
1441    fout<<"-chave:"<<endl;///<escreve o titulo chave
1442    fout<<int_chave<<endl;///<escreve a chave
1443
1444    fout<<"-data:␣"<<endl;///<escreve a string -data:
1445    fout<<"--dia:␣"<<endl;///<escreve a string --dia:
1446    cout<<"Data␣de␣Cadastro␣da␣Amostra:␣"<<endl;;
1447    repetedia:
1448
1449    cout<<"Dia:␣";
1450    while(!(cin>>num_int))///<correcao de erro caso entre alguma
            resposta errada
1451    {
1452      cin.clear();
1453      cin>>ch;
1454      cout<<endl<<"Resposta␣Incorreta!"<<endl;
1455      cin.get();
1456      goto repetedia;///<nao aceita dia maior que 31
1457    }
1458
```

```
1459    if (num_int>31)
1460    {
1461      cout<<endl<<"Resposta␣Incorreta!"<<endl;
1462      cout<<"Dia␣maior␣que␣31!"<<endl;
1463      goto repetedia;///<nao aceita dia maior que 31
1464    }
1465    fout<<num_int<<endl;
1466
1467    fout<<"--mes:␣"<<endl;///<escreve a string --dia:
1468    repetemes:
1469    cout<<"Mes:␣";
1470    while(!(cin>>num_int))///<correcao de erro caso entre alguma
            resposta errada
1471    {
1472      cin.clear();
1473      cin>>ch;
1474      cin.get();
1475      cout<<endl<<"Resposta␣Incorreta!"<<endl;
1476      goto repetemes;///<nao aceita dia maior que 31
1477    }
1478
1479    if (num_int>12)
1480    {
1481      cout<<endl<<"Resposta␣Incorreta!"<<endl;
1482      cout<<"Mes␣maior␣que␣12!"<<endl;
1483      goto repetemes;///<nao aceita dia maior que 31
1484    }
1485    fout<<num_int<<endl;
1486
1487    fout<<"--ano:␣"<<endl;///<escreve a string --dia:
1488    cout<<"Ano:␣";
1489    while(!(cin>>num_int))///<correcao de erro caso entre alguma
            resposta errada
1490    {
1491      cin.clear();
1492      cin>>ch;
1493      cout<<endl<<"Resposta␣Incorreta!"<<endl;
1494    }
1495    cin.get();
1496    fout<<num_int<<endl;
1497
1498    fout<<"-Nome_do_Fluido_de_Perfuracao:␣"<<endl;///<escreve o titulo
            bacia
1499    cout<<"Nome␣do␣Fluido␣de␣Perfuracao␣(9␣caracteres):␣";
1500    cin>>nome;
1501    cin.get();
1502    fout<<nome<<endl;
1503
```

```
1504    fout<<"-Base:␣"<<endl;///<escreve o titulo bacia
1505    cout<<"Base␣:␣";
1506    cin>>nome;
1507    cin.get();
1508    fout<<nome<<endl;
1509
1510    fout<<"-Teor_da_base:␣"<<endl;
1511
1512    cout<<"Teor␣da␣base:␣";
1513    cin>>nome;
1514    cin.get();
1515
1516    if(nome=="-")
1517    {
1518      num=999999;
1519    }
1520    else
1521    {
1522      if(atof(nome.c_str()))
1523      {
1524        num=atof(nome.c_str());
1525      }
1526      else
1527      {
1528        num=999999;
1529      }
1530    }
1531
1532    fout<<num<<endl;
1533
1534
1535    fout<<"-teor_de_agua:␣"<<endl;
1536    cout<<"Teor␣de␣Agua:␣";
1537    cin>>nome;
1538    cin.get();
1539
1540    if(nome=="-")
1541    {
1542      num=999999;
1543    }
1544    else
1545    {
1546      if(atof(nome.c_str()))
1547      {
1548        num=atof(nome.c_str());
1549      }
1550      else
1551      {
```

```
1552          num=999999;
1553        }
1554      }
1555      fout<<num<<endl;
1556
1557      fout<<"
             ------------------------------------------------------------------------
             "<<endl;
1558      fout<<"PROPRIEDADES_FISICAS_E_QUIMICAS"<<endl;
1559      fout<<"-pH_min:␣"<<endl;
1560      cout<<"pH␣minimo:␣␣";
1561      cin>>nome;
1562      cin.get();
1563
1564      if(nome=="-")
1565      {
1566        num=999999;
1567      }
1568      else
1569      {
1570        if(atof(nome.c_str()))
1571        {
1572          num=atof(nome.c_str());
1573        }
1574        else
1575        {
1576          num=999999;
1577        }
1578      }
1579      fout<<num<<endl;
1580
1581      fout<<"-pH_max:␣"<<endl;
1582      cout<<"pH␣maximo:␣";
1583      cin>>nome;
1584      cin.get();
1585
1586      if(nome=="-")
1587      {
1588        num=999999;
1589      }
1590      else
1591      {
1592        if(atof(nome.c_str()))
1593        {
1594          num=atof(nome.c_str());
1595        }
1596        else
1597        {
```

```
1598          num=999999;
1599        }
1600      }
1601      fout<<num<<endl;
1602
1603
1604      fout<<"-Peso_especifico_min:␣"<<endl;
1605      cout<<"Peso␣especifico␣minimo:␣";
1606      cin>>nome;
1607      cin.get();
1608
1609      if(nome=="-")
1610      {
1611        num=999999;
1612      }
1613      else
1614      {
1615        if(atof(nome.c_str()))
1616        {
1617          num=atof(nome.c_str());
1618        }
1619        else
1620        {
1621          num=999999;
1622        }
1623      }
1624      fout<<num<<endl;
1625
1626      fout<<"-Peso_especifico_max:␣"<<endl;
1627
1628      cout<<"Peso␣especifico␣maximo:␣";
1629      cin>>nome;
1630      cin.get();
1631
1632      if(nome=="-")
1633      {
1634        num=999999;
1635      }
1636      else
1637      {
1638        if(atof(nome.c_str()))
1639        {
1640          num=atof(nome.c_str());
1641        }
1642        else
1643        {
1644          num=999999;
1645        }
```

```
1646    }
1647    fout<<num<<endl;
1648
1649    fout<<"-Temperatura_envelhecimento:␣"<<endl;
1650
1651    cout<<"Temperatura␣do␣envelhecimento:␣";
1652    cin>>nome;
1653    cin.get();
1654
1655    if(nome=="-")
1656    {
1657      num=999999;
1658    }
1659    else
1660    {
1661      if(atof(nome.c_str()))
1662      {
1663        num=atof(nome.c_str());
1664      }
1665      else
1666      {
1667        num=999999;
1668      }
1669    }
1670    fout<<num<<endl;
1671
1672    fout<<"-Forca_gel_inicial_antes_do_envelhecimento:␣"<<endl;
1673
1674    cout<<"Forca␣gel␣inicial␣antes␣do␣envelhecimento:␣";
1675    cin>>nome;
1676    cin.get();
1677
1678    if(nome=="-")
1679    {
1680      num=999999;
1681    }
1682    else
1683    {
1684      if(atof(nome.c_str()))
1685      {
1686        num=atof(nome.c_str());
1687      }
1688      else
1689      {
1690        num=999999;
1691      }
1692    }
1693    fout<<num<<endl;
```

```
1694
1695    fout<<"-Forca_gel_final_antes_do_envelhecimento:␣"<<endl;
1696
1697    cout<<"Forca␣gel␣final␣antes␣do␣envelhecimento:␣";
1698    cin>>nome;
1699    cin.get();
1700
1701    if(nome=="-")
1702    {
1703      num=999999;
1704    }
1705    else
1706    {
1707      if(atof(nome.c_str()))
1708      {
1709        num=atof(nome.c_str());
1710      }
1711      else
1712      {
1713        num=999999;
1714      }
1715    }
1716    fout<<num<<endl;
1717
1718    fout<<"-Forca_gel_inicial_depois_do_envelhecimento:␣"<<endl;
1719
1720    cout<<"Forca␣gel␣inicial␣depois␣do␣envelhecimento:␣";
1721    cin>>nome;
1722    cin.get();
1723
1724    if(nome=="-")
1725    {
1726      num=999999;
1727    }
1728    else
1729    {
1730      if(atof(nome.c_str()))
1731      {
1732        num=atof(nome.c_str());
1733      }
1734      else
1735      {
1736        num=999999;
1737      }
1738    }
1739    fout<<num<<endl;
1740
1741    fout<<"-Forca_gel_final_depois_do_envelhecimento:␣"<<endl;
```

```
1742
1743    cout<<"Forca␣gel␣final␣depois␣do␣envelhecimento:␣";
1744    cin>>nome;
1745    cin.get();
1746
1747    if(nome=="-")
1748    {
1749      num=999999;
1750    }
1751    else
1752    {
1753      if(atof(nome.c_str()))
1754      {
1755        num=atof(nome.c_str());
1756      }
1757      else
1758      {
1759        num=999999;
1760      }
1761    }
1762    fout<<num<<endl;
1763
1764
1765    fout<<"-Estabilidade_eletrica_antes_do_envelhecimento:␣"<<endl;
1766
1767    cout<<"Estabilidade␣eletrica␣antes␣do␣envelhecimento:␣";
1768    cin>>nome;
1769    cin.get();
1770
1771    if(nome=="-")
1772    {
1773      num=999999;
1774    }
1775    else
1776    {
1777      if(atof(nome.c_str()))
1778      {
1779        num=atof(nome.c_str());
1780      }
1781      else
1782      {
1783        num=999999;
1784      }
1785    }
1786    fout<<num<<endl;
1787
1788    fout<<"-Estabilidade_eletrica_depois_do_envelhecimento:␣"<<endl;
1789
```

```
1790    cout<<"Estabilidade␣eletrica␣depois␣do␣envelhecimento:␣";
1791    cin>>nome;
1792    cin.get();
1793
1794    if(nome=="-")
1795    {
1796      num=999999;
1797    }
1798    else
1799    {
1800      if(atof(nome.c_str()))
1801      {
1802        num=atof(nome.c_str());
1803      }
1804      else
1805      {
1806        num=999999;
1807      }
1808    }
1809    fout<<num<<endl;
1810
1811    fout<<"-Coeficiente_de_lubricidade:␣"<<endl;
1812
1813    cout<<"Coeficiente␣de␣Lubricidade:␣";
1814    cin>>nome;
1815    cin.get();
1816
1817    if(nome=="-")
1818    {
1819      num=999999;
1820    }
1821    else
1822    {
1823      if(atof(nome.c_str()))
1824      {
1825        num=atof(nome.c_str());
1826      }
1827      else
1828      {
1829        num=999999;
1830      }
1831    }
1832    fout<<num<<endl;
1833
1834    fout<<"-Volume_de_Filtrado:␣"<<endl;
1835
1836    cout<<"Volume␣de␣Filtrado:␣";
1837    cin>>nome;
```

```
1838    cin.get();
1839
1840    if(nome=="-")
1841    {
1842      num=999999;
1843    }
1844    else
1845    {
1846      if(atof(nome.c_str()))
1847      {
1848        num=atof(nome.c_str());
1849      }
1850      else
1851      {
1852        num=999999;
1853      }
1854    }
1855    fout<<num<<endl;
1856
1857    fout<<"-Teor_de_solidos:␣"<<endl;
1858
1859    cout<<"Teor␣de␣solidos:␣";
1860    cin>>nome;
1861    cin.get();
1862
1863    if(nome=="-")
1864    {
1865      num=999999;
1866    }
1867    else
1868    {
1869      if(atof(nome.c_str()))
1870      {
1871        num=atof(nome.c_str());
1872      }
1873      else
1874      {
1875        num=999999;
1876      }
1877    }
1878    fout<<num<<endl;
1879
1880    fout<<"-Salinidade:␣"<<endl;
1881
1882    cout<<"Salinidade:␣";
1883    cin>>nome;
1884    cin.get();
1885
```

```
1886    if(nome=="-")
1887    {
1888      num=999999;
1889    }
1890    else
1891    {
1892      if(atof(nome.c_str()))
1893      {
1894        num=atof(nome.c_str());
1895      }
1896      else
1897      {
1898        num=999999;
1899      }
1900    }
1901    fout<<num<<endl;
1902
1903    fout<<"_____________________________________________________________
          "<<endl;
1904    fout<<"PROPRIEDADES_REOLOGICAS"<<endl;
1905    fout<<"-Viscosidade_aparente_antes_do_envelhecimento:␣"<<endl;
1906
1907    cout<<"Viscosidade␣aparente␣antes␣do␣envelhecimento:␣";
1908    cin>>nome;
1909    cin.get();
1910
1911    if(nome=="-")
1912    {
1913      num=999999;
1914    }
1915    else
1916    {
1917      if(atof(nome.c_str()))
1918      {
1919        num=atof(nome.c_str());
1920      }
1921      else
1922      {
1923        num=999999;
1924      }
1925    }
1926    fout<<num<<endl;
1927
1928    fout<<"-Viscosidade_aparente_depois_do_envelhecimento:␣"<<endl;
1929
1930    cout<<"Viscosidade␣aparente␣depois␣do␣envelhecimento:␣";
1931    cin>>nome;
1932    cin.get();
```

```
1933
1934    if(nome=="-")
1935    {
1936      num=999999;
1937    }
1938    else
1939    {
1940      if(atof(nome.c_str()))
1941      {
1942        num=atof(nome.c_str());
1943      }
1944      else
1945      {
1946        num=999999;
1947      }
1948    }
1949    fout<<num<<endl;
1950
1951    fout<<"-Viscosidade_plastica_antes_do_envelhecimento:␣"<<endl;
1952
1953    cout<<"Viscosidade␣plastica␣antes␣do␣envelhecimento:␣";
1954    cin>>nome;
1955    cin.get();
1956
1957    if(nome=="-")
1958    {
1959      num=999999;
1960    }
1961    else
1962    {
1963      if(atof(nome.c_str()))
1964      {
1965        num=atof(nome.c_str());
1966      }
1967      else
1968      {
1969        num=999999;
1970      }
1971    }
1972    fout<<num<<endl;
1973
1974    fout<<"-Viscosidade_plastica_depois_do_envelhecimento:␣"<<endl;
1975
1976    cout<<"Viscosidade␣plastica␣depois␣do␣envelhecimento:␣";
1977    cin>>nome;
1978    cin.get();
1979
1980    if(nome=="-")
```

```
1981    {
1982      num=999999;
1983    }
1984    else
1985    {
1986      if(atof(nome.c_str()))
1987      {
1988        num=atof(nome.c_str());
1989      }
1990      else
1991      {
1992        num=999999;
1993      }
1994    }
1995    fout<<num<<endl;
1996
1997    fout<<"-Limite_de_escoamento_antes_do_envelhecimento:␣"<<endl;
1998
1999    cout<<"Limite␣de␣Escoamento␣antes␣do␣envelheciemnto:␣␣";
2000    cin>>nome;
2001    cin.get();
2002
2003    if(nome=="-")
2004    {
2005      num=999999;
2006    }
2007    else
2008    {
2009      if(atof(nome.c_str()))
2010      {
2011        num=atof(nome.c_str());
2012      }
2013      else
2014      {
2015        num=999999;
2016      }
2017    }
2018    fout<<num<<endl;
2019
2020    fout<<"-Limite_de_escoamento_depois_do_envelhecimento:␣"<<endl;
2021
2022    cout<<"Limite␣de␣Escoamento␣depois␣do␣envelheciemnto:␣␣";
2023    cin>>nome;
2024    cin.get();
2025
2026    if(nome=="-")
2027    {
2028      num=999999;
```

```
2029      }
2030      else
2031      {
2032        if(atof(nome.c_str()))
2033        {
2034          num=atof(nome.c_str());
2035        }
2036        else
2037        {
2038          num=999999;
2039        }
2040      }
2041      fout<<num<<endl;
2042
2043      fout<<"_____________________________________________________________
              "<<endl;
2044      fout<<"ADITIVOS"<<endl;
2045      fout<<"-Adensante:␣"<<endl;
2046      cout<<"Adensante␣utilizado␣(9␣caracteres):␣";
2047      cin>>nome;
2048      cin.get();
2049      fout<<nome<<endl;
2050
2051      fout<<"-Concentracao_do_adensante:␣"<<endl;
2052
2053      cout<<"␣Concentracao␣do␣adensante:␣";
2054      cin>>nome;
2055      cin.get();
2056
2057      if(nome=="-")
2058      {
2059        num=999999;
2060      }
2061      else
2062      {
2063        if(atof(nome.c_str()))
2064        {
2065          num=atof(nome.c_str());
2066        }
2067        else
2068        {
2069          num=999999;
2070        }
2071      }
2072
2073      fout<<num<<endl;
2074
2075      fout<<"-Inibidor_de_formacoes_ativas:␣"<<endl;
```

```
2076    cout<<"Inibidor de formacoes ativas (9 caracteres): ";
2077    cin>>nome;
2078    cin.get();
2079    fout<<nome<<endl;
2080
2081    fout<<"-Concentração_do_inibidor_de_formacoes_ativas: "<<endl;
2082
2083    cout<<"Concentração do Inibidor de formacoes ativas: ";
2084    cin>>nome;
2085    cin.get();
2086
2087    if(nome=="-")
2088    {
2089      num=999999;
2090    }
2091    else
2092    {
2093      if(atof(nome.c_str()))
2094      {
2095        num=atof(nome.c_str());
2096      }
2097      else
2098      {
2099        num=999999;
2100      }
2101    }
2102
2103    fout<<num<<endl;
2104
2105    fout<<"-Redutor_de_filtrado: "<<endl;
2106    cout<<"Redutor de filtrado (9 caracteres): ";
2107    cin>>nome;
2108    cin.get();
2109    fout<<nome<<endl;
2110
2111    fout<<"-Concentracao_do_redutor_de_filtrado: "<<endl;
2112
2113    cout<<"Concentracao do redutor de filtrado: ";
2114    cin>>nome;
2115    cin.get();
2116
2117    if(nome=="-")
2118    {
2119      num=999999;
2120    }
2121    else
2122    {
2123      if(atof(nome.c_str()))
```

```
2124        {
2125          num=atof(nome.c_str());
2126        }
2127        else
2128        {
2129          num=999999;
2130        }
2131      }
2132
2133      fout<<num<<endl;
2134
2135      fout<<"-Biopolimero:␣"<<endl;
2136      cout<<"Biopolimero␣(9␣caracteres):␣";
2137      cin>>nome;
2138      cin.get();
2139      fout<<nome<<endl;
2140
2141      fout<<"-Concentracao_do_biopolimero:␣"<<endl;
2142
2143      cout<<"Concentracao␣do␣biopolimero␣␣:";
2144      cin>>nome;
2145      cin.get();
2146
2147      if(nome=="-")
2148      {
2149        num=999999;
2150      }
2151      else
2152      {
2153        if(atof(nome.c_str()))
2154        {
2155          num=atof(nome.c_str());
2156        }
2157        else
2158        {
2159          num=999999;
2160        }
2161      }
2162
2163      fout<<num<<endl;
2164
2165      fout<<"-Viscosificante:␣"<<endl;
2166      cout<<"Viscosificante␣(9␣caracteres):␣";
2167      cin>>nome;
2168      cin.get();
2169      fout<<nome<<endl;
2170
2171      fout<<"-Concentracao_do_viscosificante:␣"<<endl;
```

```
2172
2173    cout<<"Concentracao␣do␣viscosificante:␣";
2174    cin>>nome;
2175    cin.get();
2176
2177    if(nome=="-")
2178    {
2179      num=999999;
2180    }
2181    else
2182    {
2183      if(atof(nome.c_str()))
2184      {
2185        num=atof(nome.c_str());
2186      }
2187      else
2188      {
2189        num=999999;
2190      }
2191    }
2192
2193    fout<<num<<endl;
2194
2195    fout<<"-Dispersante:␣"<<endl;
2196    cout<<"Dispersante␣(9␣caracteres):␣";
2197    cin>>nome;
2198    cin.get();
2199    fout<<nome<<endl;
2200
2201    fout<<"-Concentracao_do_dispersante:␣"<<endl;
2202
2203    cout<<"Concentracao␣do␣dispersante:␣";
2204    cin>>nome;
2205    cin.get();
2206
2207    if(nome=="-")
2208    {
2209      num=999999;
2210    }
2211    else
2212    {
2213      if(atof(nome.c_str()))
2214      {
2215        num=atof(nome.c_str());
2216      }
2217      else
2218      {
2219        num=999999;
```

```
2220        }
2221    }
2222
2223    fout<<num<<endl;
2224
2225    fout<<"-Defloculante:␣"<<endl;
2226    cout<<"Defloculante␣(9␣caracteres):␣";
2227    cin>>nome;
2228    cin.get();
2229    fout<<nome<<endl;
2230
2231    fout<<"-Concentracao_do_defloculante:␣"<<endl;
2232
2233    cout<<"Concentracao␣do␣defloculante:␣";
2234    cin>>nome;
2235    cin.get();
2236
2237    if(nome=="-")
2238    {
2239      num=999999;
2240    }
2241    else
2242    {
2243      if(atof(nome.c_str()))
2244      {
2245        num=atof(nome.c_str());
2246      }
2247      else
2248      {
2249        num=999999;
2250      }
2251    }
2252
2253    fout<<num<<endl;
2254
2255    fout<<"-Emulsificante:␣"<<endl;
2256    cout<<"Emulsificante␣(9␣caracteres):␣";
2257    cin>>nome;
2258    cin.get();
2259    fout<<nome<<endl;
2260
2261    fout<<"-Concentracao_do_emulsificante:␣"<<endl;
2262
2263    cout<<"Concentracao␣do␣emulsificante:␣";
2264    cin>>nome;
2265    cin.get();
2266
2267    if(nome=="-")
```

```
2268    {
2269      num=999999;
2270    }
2271    else
2272    {
2273      if(atof(nome.c_str()))
2274      {
2275        num=atof(nome.c_str());
2276      }
2277      else
2278      {
2279        num=999999;
2280      }
2281    }
2282
2283    fout<<num<<endl;
2284
2285    fout<<"-Biocida:␣"<<endl;
2286    cout<<"Biocida␣(9␣caracteres):␣";
2287    cin>>nome;
2288    cin.get();
2289    fout<<nome<<endl;
2290
2291    fout<<"-Concentracao_do_biocida:␣"<<endl;
2292
2293    cout<<"Concentracao␣do␣biocida:␣";
2294    cin>>nome;
2295    cin.get();
2296
2297    if(nome=="-")
2298    {
2299      num=999999;
2300    }
2301    else
2302    {
2303      if(atof(nome.c_str()))
2304      {
2305        num=atof(nome.c_str());
2306      }
2307      else
2308      {
2309        num=999999;
2310      }
2311    }
2312
2313    fout<<num<<endl;
2314
2315    fout<<"-Lubrificante:␣"<<endl;
```

```
2316    cout<<"Lubrificante␣(9␣caracteres):␣";
2317    cin>>nome;
2318    cin.get();
2319    fout<<nome<<endl;
2320
2321    fout<<"-Concentracao_do_lubrificante:␣"<<endl;
2322
2323    cout<<"Concentracao␣do␣lubrificante:␣";
2324    cin>>nome;
2325    cin.get();
2326
2327    if(nome=="-")
2328    {
2329      num=999999;
2330    }
2331    else
2332    {
2333      if(atof(nome.c_str()))
2334      {
2335        num=atof(nome.c_str());
2336      }
2337      else
2338      {
2339        num=999999;
2340      }
2341    }
2342
2343    fout<<num<<endl;
2344
2345    fout<<"-Inibidor_de_corrosao:␣"<<endl;
2346    cout<<"Inibidor␣de␣corrosao␣(9␣caracteres):␣";
2347    cin>>nome;
2348    cin.get();
2349    fout<<nome<<endl;
2350
2351    fout<<"-Concentracao_do_inibidor_de_corrosao:␣"<<endl;
2352
2353    cout<<"Concentracao␣do␣inibidor␣de␣corrosao:␣";
2354    cin>>nome;
2355    cin.get();
2356
2357    if(nome=="-")
2358    {
2359      num=999999;
2360    }
2361    else
2362    {
2363      if(atof(nome.c_str()))
```

```
2364        {
2365          num=atof(nome.c_str());
2366        }
2367        else
2368        {
2369          num=999999;
2370        }
2371      }
2372
2373      fout<<num<<endl;
2374
2375      fout<<"-Controlador_de_pH:␣"<<endl;
2376      cout<<"Controlador␣de␣pH␣(9␣caracteres):␣";
2377      cin>>nome;
2378      cin.get();
2379      fout<<nome<<endl;
2380
2381      fout<<"-Concentracao_do_controlador_de_pH:␣"<<endl;
2382      cout<<"Concentracao␣do␣controlador␣de␣pH:␣";
2383      cin>>nome;
2384      cin.get();
2385
2386      if(nome=="-")
2387      {
2388        num=999999;
2389      }
2390      else
2391      {
2392        if(atof(nome.c_str()))
2393        {
2394          num=atof(nome.c_str());
2395        }
2396        else
2397        {
2398          num=999999;
2399        }
2400      }
2401
2402      fout<<num<<endl;
2403
2404      fout.close();
2405
2406
2407      ////////////////////////////////////
2408      //fim da inclusao do arquivo basico
2409      ////////////////////////////////////
2410  }
```

Apresenta-se na listagem 6.5 o arquivo com código da classe CConfiguracao.

Listing 6.5: Arquivo de cabeçalho da classe CConfiguracao.

```
2412#ifndef CConfiguracao_h
2413#define CConfiguracao_h
2414#include <string>
2415#include <sstream>
2416#include <iostream>
2417#include <fstream>
2418using namespace std;
2419
2420class CConfiguracao
2421{
2422public:
2423
2424  /// Representa a classe que configura o explorador e o editor de texto
2425  ifstream fin;
2426  ofstream fout;
2427
2428  stringstream caminho_ss;///<usado para conversão em string
2429  string caminho_str;///<representa o caminho do diretório em string
2430  string nome_programa;///<nome do programa definido pelo usuário
2431  string nome_explorador_linux;///<representa o explorador do linux
2432  string nome_explorador_windows;///<representa o explorador do windows
2433  string nome_editor_texto_linux;///<representa o nome do editor texto
       do linux
2434  string nome_editor_texto_windows;///<representa o nome do editor texto
        do windows
2435  string tipo_programa;///<representa o tipo de programa, se é
       explorador ou editor de texto
2436  string str;///<representa uma string
2437
2438  int opcao;///<representa a opção selecionada pelo usuário no menu
2439
2440public:
2441
2442  int opcao_abrir_DT();///<método que retorna a opção para abrir o
       diretório
2443  void opcao_abrir_DT(int);///<método que edita a opção de abrir ou não
       o diretório
2444  void opcao_programa(string, string);///<método para a troca do
       explorador e do editor de texto
2445  string opcao_programa(string);///<método para a troca do explorador
2446
2447};
2448
2449#endif
```

Apresenta-se na listagem 6.6 o arquivo de implementação da classe CConfiguracao.

Listing 6.6: Arquivo de implementação da classe CConfiguracao.

```
2450#include <iostream>
2451#include <stdio.h>
2452///verifica sistema operacional
2453#ifdef __linux
2454
2455#include <cstdlib>
2456
2457#elif _WIN32
2458
2459#include <windows.h>
2460#include <stdlib.h>///<para system
2461#include <conio.h>
2462
2463#else
2464
2465#endif
2466
2467#include <string>
2468#include <sstream>
2469#include <cstdlib>
2470#include <fstream>
2471
2472#include "CConfiguracao.h"
2473
2474using namespace std;
2475
2476int CConfiguracao::opcao_abrir_DT()
2477{
2478  caminho_ss.str("");
2479  #ifdef __linux
2480  caminho_ss<<"config"<<char(47)<<"DT.ini";
2481  #elif _WIN32
2482  caminho_ss<<"config"<<char(92)<<"DT.ini";
2483  #else
2484
2485  #endif
2486  fin.open(caminho_ss.str().c_str());
2487  if(fin.fail())
2488  {
2489    cout<<"Não␣foi␣possivel␣ler."<<endl;
2490  }
2491  caminho_ss.str("");
2492  fin>>opcao;
2493  fin.close();
2494  return opcao;
2495}
2496
```

```
2497
2498 void CConfiguracao::opcao_abrir_DT(int opcao_)
2499 {
2500   opcao=opcao_;
2501   caminho_ss.str("");
2502   #ifdef __linux
2503   caminho_ss<<"config"<<char(47)<<"DT.ini";
2504   #elif _WIN32
2505   caminho_ss<<"config"<<char(92)<<"DT.ini";
2506   #else
2507
2508   #endif
2509   fout.open(caminho_ss.str().c_str());
2510   if(fout.fail())
2511   {
2512     cout<<"Não foi possivel editar."<<endl;
2513   }
2514   caminho_ss.str("");
2515   fout<<opcao;
2516   fout.close();
2517 }
2518
2519 void CConfiguracao::opcao_programa(string tipo_programa_, string
       nome_programa_)
2520 {
2521   tipo_programa=tipo_programa_;
2522   nome_programa=nome_programa_;
2523   caminho_ss.str("");
2524   #ifdef __linux
2525   caminho_ss<<"config"<<char(47)<<"programas.ini";
2526   #elif _WIN32
2527   caminho_ss<<"config"<<char(92)<<"programas.ini";
2528   #else
2529
2530   #endif
2531   ///le o arquivo todo
2532   fin.open(caminho_ss.str().c_str());
2533   if(fin.fail())
2534   {
2535     cout<<"Não foi possivel ler."<<endl;
2536   }
2537   caminho_ss.str("");
2538   fin>>str;
2539   fin>>str;
2540   fin>>str;
2541   fin>>str;
2542
2543   fin>>nome_explorador_windows;
```

```
2544  fin>>nome_explorador_linux;
2545  fin>>str;
2546
2547  fin>>nome_editor_texto_windows;
2548  fin>>nome_editor_texto_linux;
2549
2550  fin.close();
2551
2552  caminho_ss.str("");
2553  #ifdef __linux
2554  caminho_ss<<"config"<<char(47)<<"programas.ini";
2555  #elif _WIN32
2556  caminho_ss<<"config"<<char(92)<<"programas.ini";
2557  #else
2558
2559  #endif
2560
2561  ///escreve o arquivo
2562  fout.open(caminho_ss.str().c_str());
2563  if(fout.fail())
2564  {
2565    cout<<"Não␣foi␣possivel␣editar."<<endl;
2566  }
2567  caminho_ss.str("");
2568  fout<<"Tipo-do-Programa:"<<endl;
2569  fout<<"nome-windows"<<endl;
2570  fout<<"nome-linux"<<endl;
2571  fout<<"explorador:"<<endl;
2572  if(tipo_programa=="explorador")
2573  {
2574    #ifdef __linux
2575    fout<<nome_explorador_windows<<endl;
2576    fout<<nome_programa<<endl;
2577    #elif _WIN32
2578    fout<<nome_programa<<endl;
2579    fout<<nome_explorador_linux<<endl;
2580    #else
2581
2582    #endif
2583  }
2584  else
2585  {
2586    fout<<nome_explorador_windows<<endl;
2587    fout<<nome_explorador_linux<<endl;
2588  }
2589
2590
2591
```

```
2592   fout<<"editor-texto:"<<endl;
2593   if(tipo_programa=="editortexto")
2594   {
2595     #ifdef __linux
2596     fout<<nome_editor_texto_windows<<endl;
2597     fout<<nome_programa<<endl;
2598     #elif _WIN32
2599     fout<<nome_programa<<endl;
2600     fout<<nome_editor_texto_linux<<endl;
2601     #else
2602
2603     #endif
2604   }
2605   else
2606   {
2607     fout<<nome_editor_texto_windows<<endl;
2608     fout<<nome_editor_texto_linux<<endl;
2609   }
2610
2611
2612   fout<<endl;
2613   fout<<"#OBS:␣O␣nome␣do␣programa␣é␣o␣mesmo␣que␣você␣digita␣no␣cmd␣do␣
         windows"<<endl;
2614   fout<<"#ou␣no␣terminal␣do␣linux."<<endl;
2615   fout.close();
2616
2617 }
2618
2619 string CConfiguracao::opcao_programa(string tipo_programa_)
2620 {
2621   tipo_programa=tipo_programa_;
2622   caminho_ss.str("");
2623   #ifdef __linux
2624   caminho_ss<<"config"<<char(47)<<"programas.ini";
2625   #elif _WIN32
2626   caminho_ss<<"config"<<char(92)<<"programas.ini";
2627   #else
2628
2629   #endif
2630   ///le o arquivo todo
2631   fin.open(caminho_ss.str().c_str());
2632   if(fin.fail())
2633   {
2634     cout<<"Não␣foi␣possivel␣ler."<<endl;
2635   }
2636   caminho_ss.str("");
2637   fin>>str;
2638   fin>>str;
```

```
2639   fin>>str;
2640   fin>>str;
2641
2642   fin>>nome_explorador_windows;
2643   fin>>nome_explorador_linux;
2644
2645   fin>>str;
2646
2647   fin>>nome_editor_texto_windows;
2648   fin>>nome_editor_texto_linux;
2649
2650   fin.close();
2651
2652
2653   if(tipo_programa=="explorador")
2654   {
2655     #ifdef __linux
2656     nome_programa = nome_explorador_linux;
2657     #elif _WIN32
2658     nome_programa = nome_explorador_windows;
2659     #else
2660
2661     #endif
2662   }
2663
2664   if(tipo_programa=="editortexto")
2665   {
2666     #ifdef __linux
2667     nome_programa = nome_editor_texto_linux;
2668     #elif _WIN32
2669     nome_programa = nome_editor_texto_windows;
2670     #else
2671
2672     #endif
2673   }
2674   return nome_programa;
2675
2676 }
```

Apresenta-se na listagem 6.7 o arquivo com código da classe CHeader.

Listing 6.7: Arquivo de cabeçalho da classe CHeader.

```
2677 #ifndef CHeader_h
2678 #define CHeader_h
2679 #include <iostream>
2680
2681 /// Classe que representa os cabeÃ§alhos a serem exibidos na tela.
2682 /// comentários
2683 /// comentários
```

```
2684 class CHeader
2685 {
2686 public:
2687
2688   int i;///<vetor i contagem de for
2689   int largura;///<vetor de largura do retangulo do titulo
2690 public:
2691   void header_carregando();                          ///< Animacao de
         carregamento
2692   void header_titulo();                              ///< Escreve titulo
         do programa
2693   void header_inicio();                              ///< Escreve
         subtitulo do programa
2694   void header_banco_de_dados();///<escreve subtitulo de listar amostras
2695   void header_banco_de_dados_lista_de_amostras();///<escreve subtitulo
         de amostras listadas
2696   void header_banco_de_dados_exibir_amostras();///<escreve subtitulo de
         exibir amostras
2697   void header_banco_de_dados_pesquisa_de_amostras();///<escreve
         subtitulo de amostras listadas
2698   void header_banco_de_dados_inserir_amostras();///<escreve subtitulo de
          inserir amostras
2699   void header_banco_de_dados_excluir_amostras();///<escreve subtitulo de
          excluir amostras
2700   void header_banco_de_dados_exportar_amostras();///<escreve subtitulo
         de excluir amostras
2701   void header_configuracoes();///<escreve subtitulo do menu 2
2702   void header_menu_abre_DT();///<escreve subtitulo do menu
2703   void header_menu_abre_bd();///<escreve subtitulo do menu
2704
2705
2706
2707 };
2708
2709
2710 #endif
```

Apresenta-se na listagem 6.8 o arquivo de implementação da classe CHeader.

Listing 6.8: Arquivo de implementação da classe CHeader.

```
2711 ///verifica sistema operacional
2712 #ifdef __linux
2713 #include <cstdlib>
2714 ///  #include <unistd.h>
2715 #elif _WIN32
2716
2717 #include <windows.h>
2718 #include <stdlib.h>///<para system
2719 #include <conio.h>
```

```
2720#else
2721#endif
2722#include <iostream>
2723#include <stdio.h>
2724#include <locale>
2725#include "CHeader.h"
2726using namespace std;
2727
2728void CHeader::header_titulo()///<escreve titulo do programa
2729{
2730
2731  ///Titulo Programa
2732  ///verifica sistema operacional
2733  #ifdef __linux
2734  ///system("clear");
2735
2736  {
2737    largura=45;
2738    cout<<"\u2554";///<canto sup esq
2739    for(i=0;i<largura;i++)///<barra sup
2740      cout<<"\u2550";
2741    cout<<"\u2557"<<endl;///<canto su dir
2742    cout<<"\u2551"<<"                                                    "<<"\
       u2551"<<endl;;///<Margem vert acim Titulo
2743    cout<<"\u2551";///<Margem vert Titulo
2744    cout<<"    Banco de Dados de Fluidos de Perfuracao    ";
2745    cout<<"\u2551"<<endl;///<Margem vert Titulo
2746    cout<<"\u255A";///<canto inf dir
2747    for(i=0;i<largura;i++)///<barra inf
2748      cout<<"\u2550";
2749    cout<<"\u255D"<<endl;///<canto inf esq
2750  }
2751  #elif _WIN32
2752  ///<system("cls");
2753
2754  {
2755
2756    largura=30;
2757    cout<<(char)201;///<canto sup esq
2758    for(i=0;i<largura;i++)///<barra sup
2759      cout<<(char)205;
2760    cout<<(char)187<<endl;///<canto su dir
2761    cout<<(char)186<<"                                                    "<<(
       char)186<<endl;;///<Margem vert acim Titulo
2762    cout<<(char)186;///<Margem vert Titulo
2763    cout<<"    Banco de Dados de Fluidos de Perfuração    ";
2764    cout<<endl;
2765    cout<<(char)186<<endl;///<Margem vert Titulo
```

```
2766      cout<<(char)200;///<canto inf dir
2767      for(i=0;i<largura;i++)///<barra inf
2768        cout<<(char)205;
2769      cout<<(char)188<<endl;///<canto inf esq
2770    }
2771
2772
2773
2774
2775
2776    #else
2777
2778    #endif
2779
2780
2781  }
2782
2783
2784
2785      void CHeader::header_inicio()///<escreve subtitulo do menu inicio
2786      {
2787        #ifdef __linux
2788        {
2789
2790          largura=30;
2791          cout<<"\u250C";///<canto sup esq
2792          for(i=0;i<largura;i++)///<barra sup
2793            cout<<"\u2500";
2794          cout<<"\u2510"<<endl;///<canto su dir
2795          cout<<"\u2502";///<Margem vert Titulo
2796          cout<<"␣␣␣␣␣␣␣␣␣␣␣␣␣␣In\u00EDcio␣␣␣␣␣␣␣␣␣␣␣␣␣␣";
2797          cout<<"\u2502"<<endl;///<Margem vert Titulo
2798          cout<<"\u2514";///<canto inf dir
2799          for(i=0;i<largura;i++)///<barra inf
2800            cout<<"\u2500";
2801          cout<<"\u2518"<<endl;
2802          cout<<endl;
2803
2804        }
2805
2806
2807
2808
2809
2810        #elif _WIN32
2811        {
2812          largura=30;
2813          cout<<(char)218;///<canto sup esq
```

```
2814          for(i=0;i<largura;i++)///<barra sup
2815            cout<<(char)196;
2816          cout<<(char)191<<endl;///<canto su dir
2817          cout<<(char)179;///<Margem vert Titulo
2818          cout<<"␣␣␣␣␣␣␣␣␣␣␣␣␣Início␣␣␣␣␣␣␣␣␣␣␣␣␣";
2819          cout<<(char)179<<endl;///<Margem vert Titulo
2820          cout<<(char)192;///<canto inf dir
2821          for(i=0;i<largura;i++)///<barra inf
2822            cout<<(char)196;
2823          cout<<(char)217<<endl;
2824          cout<<endl;
2825
2826
2827        }
2828        #else
2829
2830        #endif
2831
2832      }
2833
2834      void CHeader::header_banco_de_dados()///<escreve subtitulo para
            banco de dados
2835      {
2836        #ifdef __linux
2837        {
2838          largura=30;
2839          cout<<"\u250C";///<canto sup esq
2840          for(i=0;i<largura;i++)///<barra sup
2841            cout<<"\u2500";
2842          cout<<"\u2510"<<endl;///<canto su dir
2843          cout<<"\u2502";///<Margem vert Titulo
2844          cout<<"␣␣␣␣␣␣␣␣Banco␣de␣Dados␣␣␣␣␣␣␣␣";
2845          cout<<"\u2502"<<endl;///<Margem vert Titulo
2846          cout<<"\u2514";///<canto inf dir
2847          for(i=0;i<largura;i++)///<barra inf
2848            cout<<"\u2500";
2849          cout<<"\u2518"<<endl;
2850          cout<<endl;
2851
2852        }
2853        #elif _WIN32
2854        {
2855          largura=30;
2856          cout<<(char)218;///<canto sup esq
2857          for(i=0;i<largura;i++)///<barra sup
2858            cout<<(char)196;
2859          cout<<(char)191<<endl;///<canto su dir
2860          cout<<(char)179;///<Margem vert Titulo
```

```
2861
2862        cout<<"␣␣␣␣␣␣␣␣Banco␣de␣Dados␣␣␣␣␣␣␣␣";
2863
2864        cout<<(char)179<<endl;///<Margem vert Titulo
2865        cout<<(char)192;///<canto inf dir
2866        for(i=0;i<largura;i++)///<barra inf
2867          cout<<(char)196;
2868        cout<<(char)217<<endl;
2869        cout<<endl;
2870      }
2871
2872      #else
2873
2874      #endif
2875
2876    }
2877
2878    void CHeader::header_banco_de_dados_lista_de_amostras()///<escreve
         subtitulo para lista de fluidos
2879    {
2880      #ifdef __linux
2881
2882      {
2883        largura=40;
2884        cout<<"\u250C";///<canto sup esq
2885        for(i=0;i<largura;i++)///<barra sup
2886          cout<<"\u2500";
2887        cout<<"\u2510"<<endl;///<canto su dir
2888        cout<<"\u2502";///<Margem vert Titulo
2889        cout<<"␣␣␣␣␣␣Lista␣de␣Fluidos␣de␣Perfuracao␣␣␣␣";
2890        cout<<"\u2502"<<endl;///<Margem vert Titulo
2891        cout<<"\u2514";///<canto inf dir
2892        for(i=0;i<largura;i++)///<barra inf
2893          cout<<"\u2500";
2894        cout<<"\u2518"<<endl;
2895        cout<<endl;
2896      }
2897
2898      #elif _WIN32
2899
2900      {
2901        largura=30;
2902        cout<<(char)218;///<canto sup esq
2903        for(i=0;i<largura;i++)///<barra sup
2904          cout<<(char)196;
2905        cout<<(char)191<<endl;///<canto su dir
2906        cout<<(char)179;///<Margem vert Titulo
2907        cout<<"␣␣␣␣␣␣Lista␣de␣Fluidos␣de␣Perfuração␣␣␣␣␣␣␣";
```

```
2908          cout<<(char)179<<endl;///<Margem vert Titulo
2909          cout<<(char)192;///<canto inf dir
2910          for(i=0;i<largura;i++)///<barra inf
2911            cout<<(char)196;
2912          cout<<(char)217<<endl;
2913          cout<<endl;
2914        }
2915
2916
2917        #else
2918
2919        #endif
2920      }
2921
2922      void CHeader::header_banco_de_dados_exibir_amostras()///<escreve
          subtitulo das amostras listadas
2923      {
2924        #ifdef __linux
2925
2926
2927        largura=30;
2928        cout<<"\u250C";///<canto sup esq
2929        for(i=0;i<largura;i++)///<barra sup
2930          cout<<"\u2500";
2931        cout<<"\u2510"<<endl;///<canto su dir
2932        cout<<"\u2502";///<Margem vert Titulo
2933
2934
2935        cout<<"␣␣␣␣␣␣␣␣␣␣␣␣␣Exibir␣␣␣␣␣␣␣␣␣␣␣␣␣";
2936
2937        cout<<"\u2502"<<endl;///<Margem vert Titulo
2938        cout<<"\u2514";///<canto inf dir
2939        for(i=0;i<largura;i++)///<barra inf
2940          cout<<"\u2500";
2941        cout<<"\u2518"<<endl;
2942        cout<<endl;
2943
2944
2945        #elif _WIN32
2946
2947        {
2948          largura=30;
2949          cout<<(char)218;///<canto sup esq
2950          for(i=0;i<largura;i++)///<barra sup
2951            cout<<(char)196;
2952          cout<<(char)191<<endl;///<canto su dir
2953          cout<<(char)179;///<Margem vert Titulo
2954
```

```
2955          cout<<"␣␣␣␣␣␣␣␣␣␣␣␣␣Exibir␣␣␣␣␣␣␣␣␣␣␣␣␣";
2956
2957
2958          cout<<(char)179<<endl;///<Margem vert Titulo
2959          cout<<(char)192;///<canto inf dir
2960          for(i=0;i<largura;i++)///<barra inf
2961            cout<<(char)196;
2962          cout<<(char)217<<endl;
2963          cout<<endl;
2964        }
2965
2966        #else
2967
2968        #endif
2969
2970      }
2971
2972      void CHeader::header_banco_de_dados_pesquisa_de_amostras()///<
           escreve subtitulo para pesquisa de amostras
2973      {
2974
2975        #ifdef __linux
2976
2977        {
2978          largura=40;
2979          cout<<"\u250C";///<canto sup esq
2980          for(i=0;i<largura;i++)///<barra sup
2981            cout<<"\u2500";
2982          cout<<"\u2510"<<endl;///<canto su dir
2983          cout<<"\u2502";///<Margem vert Titulo
2984
2985          cout<<"␣␣␣␣␣Pesquisa␣de␣Fluidos␣de␣Perfuracao␣␣␣";
2986
2987          cout<<"\u2502"<<endl;///<Margem vert Titulo
2988          cout<<"\u2514";///<canto inf dir
2989          for(i=0;i<largura;i++)///<barra inf
2990            cout<<"\u2500";
2991          cout<<"\u2518"<<endl;
2992          cout<<endl;
2993        }
2994        #elif _WIN32
2995        {
2996          largura=30;
2997          cout<<(char)218;///<canto sup esq
2998          for(i=0;i<largura;i++)///<barra sup
2999            cout<<(char)196;
3000          cout<<(char)191<<endl;///<canto su dir
3001          cout<<(char)179;///<Margem vert Titulo
```

```
3002
3003        cout<<"␣␣␣␣␣Pesquisa␣de␣Fluidos␣de␣Perfuração␣␣␣";
3004
3005        cout<<(char)179<<endl;///<Margem vert Titulo
3006        cout<<(char)192;///<canto inf dir
3007        for(i=0;i<largura;i++)///<barra inf
3008          cout<<(char)196;
3009        cout<<(char)217<<endl;
3010        cout<<endl;
3011      }
3012
3013      #else
3014
3015      #endif
3016
3017    }
3018    void CHeader::header_banco_de_dados_inserir_amostras()///<escreve
         subtitulo para inserir fluidos
3019    {
3020
3021      #ifdef __linux
3022
3023      largura=40;
3024      cout<<"\u250C";///<canto sup esq
3025      for(i=0;i<largura;i++)///<barra sup
3026        cout<<"\u2500";
3027      cout<<"\u2510"<<endl;///<canto su dir
3028      cout<<"\u2502";///<Margem vert Titulo
3029
3030      cout<<"␣␣␣␣␣␣␣Inserir␣Fluidos␣de␣Perfuracao␣␣␣␣";
3031
3032      cout<<"\u2502"<<endl;///<Margem vert Titulo
3033      cout<<"\u2514";///<canto inf dir
3034      for(i=0;i<largura;i++)///<barra inf
3035        cout<<"\u2500";
3036      cout<<"\u2518"<<endl;
3037      cout<<endl;
3038
3039      #elif _WIN32
3040
3041      {
3042        largura=30;
3043        cout<<(char)218;///<canto sup esq
3044        for(i=0;i<largura;i++)///<barra sup
3045          cout<<(char)196;
3046        cout<<(char)191<<endl;///<canto su dir
3047        cout<<(char)179;///<Margem vert Titulo
3048
```

```
3049          cout<<"␣␣␣␣␣␣␣Inserir␣Fluidos␣de␣Perfuracao␣␣␣";
3050
3051          cout<<(char)179<<endl;///<Margem vert Titulo
3052          cout<<(char)192;///<canto inf dir
3053          for(i=0;i<largura;i++)///<barra inf
3054            cout<<(char)196;
3055          cout<<(char)217<<endl;
3056          cout<<endl;
3057        }
3058
3059        #else
3060
3061        #endif
3062      }
3063
3064      void CHeader::header_banco_de_dados_excluir_amostras()///<escreve
           subtitulo das amostras listadas
3065      {
3066
3067        #ifdef __linux
3068
3069        largura=30;
3070        cout<<"\u250C";///<canto sup esq
3071        for(i=0;i<largura;i++)///<barra sup
3072          cout<<"\u2500";
3073        cout<<"\u2510"<<endl;///<canto su dir
3074        cout<<"\u2502";///<Margem vert Titulo
3075
3076
3077        cout<<"␣␣␣␣␣␣␣Excluir␣Amostras␣␣␣␣␣␣␣";
3078
3079        cout<<"\u2502"<<endl;///<Margem vert Titulo
3080        cout<<"\u2514";///<canto inf dir
3081        for(i=0;i<largura;i++)///<barra inf
3082          cout<<"\u2500";
3083        cout<<"\u2518"<<endl;
3084        cout<<endl;
3085
3086        #elif _WIN32
3087
3088        {
3089          largura=30;
3090          cout<<(char)218;///<canto sup esq
3091          for(i=0;i<largura;i++)///<barra sup
3092            cout<<(char)196;
3093          cout<<(char)191<<endl;///<canto su dir
3094          cout<<(char)179;///<Margem vert Titulo
3095          cout<<"␣␣␣␣␣␣␣Excluir␣Amostras␣␣␣␣␣␣␣";
```

```
3096          cout<<(char)179<<endl;///<Margem vert Titulo
3097          cout<<(char)192;///<canto inf dir
3098          for(i=0;i<largura;i++)///<barra inf
3099            cout<<(char)196;
3100          cout<<(char)217<<endl;
3101          cout<<endl;
3102        }
3103
3104        #else
3105
3106        #endif
3107
3108
3109      }
3110
3111
3112      void CHeader::header_banco_de_dados_exportar_amostras()///<escreve
            subtitulo para exportar
3113      {
3114        #ifdef __linux
3115        {
3116
3117          largura=40;
3118          cout<<"\u250C";///<canto sup esq
3119          for(i=0;i<largura;i++)///<barra sup
3120            cout<<"\u2500";
3121          cout<<"\u2510"<<endl;///<canto su dir
3122          cout<<"\u2502";///<Margem vert Titulo
3123
3124          cout<<"␣␣␣␣␣␣␣␣Exportar␣Fluidos␣de␣Perfuracao␣␣␣␣";
3125
3126          cout<<"\u2502"<<endl;///<Margem vert Titulo
3127          cout<<"\u2514";///<canto inf dir
3128          for(i=0;i<largura;i++)///<barra inf
3129            cout<<"\u2500";
3130          cout<<"\u2518"<<endl;
3131          cout<<endl;
3132        }
3133        #elif _WIN32
3134
3135        {
3136          largura=30;
3137          cout<<(char)218;///<canto sup esq
3138          for(i=0;i<largura;i++)///<barra sup
3139            cout<<(char)196;
3140          cout<<(char)191<<endl;///<canto su dir
3141          cout<<(char)179;///<Margem vert Titulo
3142
```

```
3143          cout<<"␣␣␣␣␣␣␣␣Exportar␣Fluidos␣de␣Perfuracao␣␣␣␣";
3144
3145          cout<<(char)179<<endl;///<Margem vert Titulo
3146          cout<<(char)192;///<canto inf dir
3147          for(i=0;i<largura;i++)///<barra inf
3148            cout<<(char)196;
3149          cout<<(char)217<<endl;
3150          cout<<endl;
3151        }
3152
3153
3154        #else
3155
3156        #endif
3157
3158      }
3159
3160
3161
3162      void CHeader::header_configuracoes()///<subtitulo do menu 2
3163      {
3164
3165
3166        #ifdef __linux
3167
3168        largura=30;
3169        cout<<"\u250C";///<canto sup esq
3170        for(i=0;i<largura;i++)///<barra sup
3171          cout<<"\u2500";
3172        cout<<"\u2510"<<endl;///<canto su dir
3173        cout<<"\u2502";///<Margem vert Titulo
3174        cout<<"␣␣␣␣␣␣␣␣␣Configura\u00E7\u00F5es␣␣␣␣␣␣␣␣␣";
3175        cout<<"\u2502"<<endl;///<Margem vert Titulo
3176        cout<<"\u2514";///<canto inf dir
3177        for(i=0;i<largura;i++)///<barra inf
3178          cout<<"\u2500";
3179        cout<<"\u2518"<<endl;
3180        cout<<endl;
3181
3182        #elif _WIN32
3183        {
3184          largura=30;
3185          cout<<(char)218;///<canto sup esq
3186          for(i=0;i<largura;i++)///<barra sup
3187            cout<<(char)196;
3188          cout<<(char)191<<endl;///<canto su dir
3189          cout<<(char)179;///<Margem vert Titulo
3190
```

```
3191          cout<<"␣␣␣␣␣␣␣␣␣Configurações␣␣␣␣␣␣␣␣␣";
3192
3193          cout<<(char)179<<endl;///<Margem vert Titulo
3194          cout<<(char)192;///<canto inf dir
3195          for(i=0;i<largura;i++)///<barra inf
3196            cout<<(char)196;
3197          cout<<(char)217<<endl;
3198          cout<<endl;
3199        }
3200
3201
3202        #else
3203
3204        #endif
3205      }
3206
3207      void CHeader::header_menu_abre_DT()///<subtitulo do menu 3
3208      {
3209        #ifdef __linux
3210
3211        largura=30;
3212        cout<<"\u250C";///<canto sup esq
3213        for(i=0;i<largura;i++)///<barra sup
3214          cout<<"\u2500";
3215        cout<<"\u2510"<<endl;///<canto su dir
3216        cout<<"\u2502";///<Margem vert Titulo
3217
3218        cout<<"␣Abrindo␣Diret\u00F3rio␣de␣Trablho␣";
3219
3220        cout<<"\u2502"<<endl;///<Margem vert Titulo
3221        cout<<"\u2514";///<canto inf dir
3222        for(i=0;i<largura;i++)///<barra inf
3223          cout<<"\u2500";
3224        cout<<"\u2518"<<endl;
3225        cout<<endl;
3226
3227        #elif _WIN32
3228        {
3229          largura=30;
3230          cout<<(char)218;///<canto sup esq
3231          for(i=0;i<largura;i++)///<barra sup
3232            cout<<(char)196;
3233          cout<<(char)191<<endl;///<canto su dir
3234          cout<<(char)179;///<Margem vert Titulo
3235
3236          cout<<"␣Abrindo␣Diretório␣de␣Trablho␣";
3237
3238          cout<<(char)179<<endl;///<Margem vert Titulo
```

```
3239          cout<<(char)192;///<canto inf dir
3240          for(i=0;i<largura;i++)///<barra inf
3241            cout<<(char)196;
3242          cout<<(char)217<<endl;
3243          cout<<endl;
3244        }
3245
3246
3247        #else
3248
3249        #endif
3250      }
3251
3252      void CHeader::header_menu_abre_bd()///<subtitulo do menu 3
3253      {
3254
3255        #ifdef __linux
3256
3257        largura=30;
3258        cout<<"\u250C";///<canto sup esq
3259        for(i=0;i<largura;i++)///<barra sup
3260          cout<<"\u2500";
3261        cout<<"\u2510"<<endl;///<canto su dir
3262        cout<<"\u2502";///<Margem vert Titulo
3263
3264        cout<<"␣␣Abrindo␣Diret\u00F3rio␣de␣Dados␣␣␣";
3265
3266        cout<<"\u2502"<<endl;///<Margem vert Titulo
3267        cout<<"\u2514";///<canto inf dir
3268        for(i=0;i<largura;i++)///<barra inf
3269          cout<<"\u2500";
3270        cout<<"\u2518"<<endl;
3271        cout<<endl;
3272
3273        #elif _WIN32
3274        {
3275          largura=30;
3276          cout<<(char)218;///<canto sup esq
3277          for(i=0;i<largura;i++)///<barra sup
3278            cout<<(char)196;
3279          cout<<(char)191<<endl;///<canto su dir
3280          cout<<(char)179;///<Margem vert Titulo
3281
3282          cout<<"␣␣Abrindo␣Diretório␣de␣Dados␣␣";
3283
3284          cout<<(char)179<<endl;///<Margem vert Titulo
3285          cout<<(char)192;///<canto inf dir
3286          for(i=0;i<largura;i++)///<barra inf
```

```
3287            cout<<(char)196;
3288          cout<<(char)217<<endl;
3289          cout<<endl;
3290        }
3291
3292
3293
3294        #else
3295
3296        #endif
3297      }
```

Apresenta-se na listagem 6.9 o arquivo com código da classe CInterface.

Listing 6.9: Arquivo de cabeçalho da classe CInterface.

```
3300#ifndef CInterface_h
3301#define CInterface_h
3302#include <iostream>
3303
3304#ifdef __linux
3305
3306#include <cstdlib>
3307
3308#elif _WIN32
3309#include <windows.h>///<para configurar a janela
3310#include <stdlib.h>///<para system
3311#else
3312
3313#endif
3314
3315#include "CMenu.h"
3316#include "CHeader.h"
3317#include "CBancodeDados.h"
3318#include "CFluidodePerfuracao.h"
3319#include "CRelatorio.h"
3320#include "CConfiguracao.h"
3321
3322
3323#include <string>
3324#include <sstream>
3325#include <vector>
3326
3327class CInterface
3328{
3329public:
3330  int resposta;///<representa a resposta ao usuário referente ao menu de
        opções
3331  int int_chave;///<representa a chave das amostras
3332  int item;///<representa os títulos do arquivo
```

```
3333   unsigned int c;///<representa um contador
3334
3335   CHeader Head;
3336   CMenu Menu;
3337   CBancodeDados bancodedados;
3338   CRelatorio relatorio;
3339   CConfiguracao configuracao;
3340
3341   stringstream comando_abrir;///<para conversão em string
3342   stringstream titulo_ss;///<para conversão em string
3343   string titulo;///<representa o título
3344   string informacao_str;///<representa o novo programa lido
3345   string programa;///<representa o nome do explorador e do editor de
          texto
3346
3347   char ch;///<representa um caracter
3348
3349   vector<CFluidodePerfuracao> amostras;///<representa o vetor de
          amostras do tipo CFluidodePerfuracao
3350
3351 public:
3352   void inicia();///<método para iniciar o programa
3353   void abre_DT();///<método para abrir diretório de trabalho
3354   void abre_bd();///<método para abrir pasta do banco de dados
3355   void abre_relatorio(int);///<método para abrir relatórios
3356   void ajusta_janela();///<método para ajustar janela
3357   void limpa_janela();///<método para limpar janela
3358
3359
3360 };
3361
3362 #endif
```

Apresenta-se na listagem 6.10 o arquivo de implementação da classe CInterface.

Listing 6.10: Arquivo de implementação da classe CInterface.

```
3363 #include <iostream>
3364 ///verifica sistema operacional
3365 #ifdef __linux
3366     #include <cstdlib>
3367     #include <stdio.h>
3368    /// #include <unistd.h>
3369 #elif _WIN32
3370         #include <windows.h>///<para configurar a janela
3371         #include <stdlib.h>///<para system
3372 #else
3373
3374 #endif
3375
```

```
3376 #include <sstream>
3377 #include <cstring>///<biblioteca para trab com os comandos
3378 #include <string>
3379 #include "CInterface.h"///<inclui classe
3380 #include "CMenu.h"///<Inclui classes utilizadas
3381 #include "CHeader.h"
3382 #include "CFluidodePerfuracao.h"
3383
3384
3385 using namespace std;
3386
3387 ///usuario simples
3388
3389 void CInterface::inicia()///<inicia o menu
3390 {
3391
3392
3393   ajusta_janela();
3394   do{
3395
3396     limpa_janela();
3397     Head.header_titulo();///<Coloca titulo principal do programa
3398     Head.header_inicio();///<Coloca subtitulo Inicio
3399     resposta=Menu.menu_inicio();///<Escreve o menu inicio e obtem a
          resposta
3400
3401     switch (resposta)///<testa a resposta
3402     {
3403       case 1:///<Banco de dados
3404         do
3405         {
3406           limpa_janela();
3407           Head.header_titulo();///<Coloca titulo pricipal do programa
3408           Head.header_banco_de_dados();///<Coloca subtitulo editar
                amostras
3409           resposta=Menu.menu_banco_de_dados(); ///<obtem resposta do
                menu editar amostra
3410
3411           switch (resposta)///<faz a funçao respctiva do submenu editar
                amostra
3412           {
3413             case 1:///<listar
3414               limpa_janela();
3415               Head.header_titulo();
3416               Head.header_banco_de_dados_lista_de_amostras();
3417               amostras=bancodedados.ler_amostra_basico();
3418               bancodedados.listar_amostra(amostras);
3419               cout<<endl;
```

```
3420                amostras.clear();
3421                  do
3422                  {
3423                  cout<<endl;
3424                  resposta=Menu.menu_exibir_voltar();
3425                  if(resposta==0)break;
3426                      do
3427                      {
3428                      limpa_janela();
3429                      Head.header_titulo();
3430                      Head.header_banco_de_dados_exibir_amostras();
3431                      amostras=bancodedados.ler_amostra_basico();
3432                      bancodedados.listar_amostra(amostras);
3433                      amostras.clear();
3434                      cout<<"CHAVE: ";
3435                      cin>>int_chave;
3436                      ///<limpa tudo e reescreve
3437                      limpa_janela();
3438                      Head.header_titulo();
3439                      Head.header_banco_de_dados_exibir_amostras();
3440                      amostras=bancodedados.ler_amostra_basico();
3441                      bancodedados.listar_amostra(amostras);
3442                      amostras.clear();
3443                      bancodedados.exibir_amostra(int_chave);
3444                      cout<<endl;
3445                      cout<<"Exibir Outra Amostra?";
3446                      resposta=Menu.menu_sim_nao();
3447                      }while(resposta!=0);
3448                  }while(resposta!=0);
3449                ///<exibe lista pergunta chave para visualizaca
3450
3451                resposta=2;
3452                break;
3453
3454              case 2:///<pesquisar
3455                limpa_janela();
3456                Head.header_titulo();
3457                Head.header_banco_de_dados_pesquisa_de_amostras();
3458                resposta=Menu.menu_banco_de_dados_pesquisar();
3459                if(resposta==2)
3460                {
3461                  resposta=1;
3462                  break;
3463                }
3464
3465                switch(resposta)
3466                {
3467
```

```
3468                case 1:
3469
3470                  do
3471                  {
3472                    limpa_janela();
3473                    Head.header_banco_de_dados_pesquisa_de_amostras();
3474                    amostras=bancodedados.ler_amostra_basico();///<lista
                            todas as amostras
3475                    cout<<"Itens de Pesquisa"<<endl<<endl;
3476                    resposta=Menu.menu_filtro();///<escreve menu filtro
                          e retorna seu valor
3477                    if(resposta==16)break;///<volta
3478                    limpa_janela();
3479                    Head.header_banco_de_dados_pesquisa_de_amostras();
3480                    amostras=bancodedados.ler_amostra_basico(resposta);
                          ///<vetor de amostra recebe amostra ja filtradas
3481                    bancodedados.listar_amostra(amostras,resposta);
3482                    cout<<endl;
3483                    resposta=Menu.menu_exibir_voltar();
3484                    if(resposta==0)break;
3485
3486
3487                    do
3488                    {
3489                      ///<limpa tudo e reescreve
3490                      limpa_janela();
3491                      Head.header_titulo();
3492                      Head.header_banco_de_dados_exibir_amostras();
3493                      ///bancodedados.listar_amostra(amostras,resposta);
3494                      ///cout<<endl;
3495                      ///resposta=Menu.menu_exibir_voltar();
3496                      ///if(resposta==0)break;
3497                      ///inicia exibicao amostra
3498                      cout<<"CHAVE: ";
3499                      cin>>int_chave;
3500                      limpa_janela();///acha a chave digitada e
                            reescreve tudo novamente
3501                      Head.header_titulo();
3502                      Head.header_banco_de_dados_exibir_amostras();
3503                      bancodedados.listar_amostra(amostras,resposta);
3504                      bancodedados.exibir_amostra(int_chave);
3505                      cout<<"Deseja Pesquisar Novamente?";
3506                      resposta=Menu.menu_sim_nao();
3507                      amostras.clear();
3508
3509                    }
3510                    while(resposta==1);
3511                  }
```

```
3512                 while(resposta==1);
3513                 break;
3514               }
3515               resposta=2;
3516               break;///<break da pesquisa
3517
3518               case 3:///<inserir
3519                 limpa_janela();
3520                 Head.header_titulo();
3521                 Head.header_banco_de_dados_inserir_amostras();
3522                 if(configuracao.opcao_abrir_DT()==1)
3523                 {
3524                   abre_DT();
3525                 }
3526
3527                 do
3528                 {
3529                   bancodedados.inserir_amostra();
3530                   cout<<endl;
3531                   cout<<endl;
3532                   cout<<"Deseja Inserir Novamente?"<<endl;
3533                   resposta=Menu.menu_sim_nao();
3534                 }
3535                 while(resposta==1);
3536             resposta=2;///<retorna ao menu Banco de DADOS
3537
3538             break;
3539
3540                case 4:///<excluir
3541                  do
3542                  {
3543                    limpa_janela();
3544                    Head.header_titulo();
3545                    Head.header_banco_de_dados_excluir_amostras();
3546                    amostras=bancodedados.ler_amostra_basico();
3547                    bancodedados.listar_amostra(amostras);
3548                    cout<<"Deseja realmente Excluir?";
3549                    resposta=Menu.menu_sim_nao();
3550
3551                    if(resposta==0)break;
3552                    cout<<"CHAVE: ";
3553
3554                    cin>>int_chave;
3555                    ///<cin.get();
3556
3557                    ///system("pause");
3558                    bancodedados.excluir_amostra(int_chave);
3559                    amostras.clear();
```

```
3560                    cout<<endl;
3561                    cout<<endl;
3562
3563                    ///atualiza tela
3564                    limpa_janela();
3565                    Head.header_titulo();
3566                    Head.header_banco_de_dados_excluir_amostras();
3567                    amostras=bancodedados.ler_amostra_basico();
3568                    bancodedados.listar_amostra(amostras);
3569                    amostras.clear();
3570                    cout<<"Deseja Excluir Outra Amostra?"<<endl;
3571                    resposta=Menu.menu_sim_nao();
3572                  }
3573                  while(resposta==1);
3574                  resposta=2;///<retorna ao menu Banco de DADOS
3575
3576
3577                  break;
3578
3579
3580              case 5:///<Exportar
3581                limpa_janela();
3582                Head.header_titulo();
3583                Head.header_banco_de_dados_exportar_amostras();
3584                resposta=Menu.menu_exportar_dados();
3585                if(resposta==2)
3586                {
3587                  resposta=2;
3588                  break;
3589                }
3590                switch(resposta)
3591                {
3592                  case 1:
3593
3594                    cout<<"CHAVE: ";
3595                    cin>>int_chave;
3596                    cin.get();
3597                    relatorio.exportar_amostra(int_chave);
3598                    abre_relatorio(int_chave);
3599                    break;
3600
3601                  default:
3602                    resposta=2;
3603                    break;
3604                    ///abre diretorio com relatorio(os)
3605                }
3606                resposta=2;
3607                break;
```

```
3608
3609                case 6:///<voltar
3610                  resposta=1;
3611                  break;
3612
3613                   default:
3614                  {
3615                    #ifdef __linux
3616                    setlocale(LC_ALL, "Portuguese");
3617                    cout<<"Op\u00E7\u00E3o invalida."<<endl;
3618                    locale::global(locale(""));
3619                    #elif _WIN32
3620                    setlocale(LC_ALL, "Portuguese");
3621                    cout<<"Opcão invalida."<<endl;
3622                    locale::global(locale(""));
3623                    #else
3624
3625                    #endif
3626
3627                    limpa_janela();
3628
3629                  }
3630                  resposta=1;
3631                  break;
3632                 }
3633          }
3634          while(resposta==2);
3635
3636          break;///<fechando o caso 1:Banco de dados
3637
3638
3639        case 2:///<configuracoes
3640          limpa_janela();
3641          Head.header_titulo();
3642          Head.header_configuracoes();
3643          resposta=Menu.menu_configuracoes();
3644
3645          switch(resposta)
3646          {
3647            case 1:
3648              cout<<endl;
3649              cout<<"Explorador Padr\u00E3o: "<<configuracao.
                     opcao_programa("explorador")<<endl;
3650              cout<<"Explorador Padr\u00E3o: ";
3651              cin>>informacao_str;
3652              cin.get();
3653              configuracao.opcao_programa("explorador",informacao_str);
3654              break;
```

```
3655
3656              case 2:
3657                cout<<endl;
3658                cout<<"Editor␣de␣Texto␣Padr\u00E3o:␣"<<configuracao.
                       opcao_programa("editortexto")<<endl;
3659                cout<<"Editor␣de␣Texto␣Padr\uE003o:␣";
3660                cin>>informacao_str;
3661                cin.get();
3662                configuracao.opcao_programa("editortexto",informacao_str);
3663                break;
3664
3665              case 3:
3666                cout<<endl;
3667                cout<<"Op\u00E7\u00E3o␣Atual:␣"<<configuracao.
                       opcao_abrir_DT()<<endl;
3668                cout<<endl;
3669                cout<<"Deseja␣sempre␣abrir␣o␣Diret\u00F3rio␣de␣Trabalho␣(
                       DT)?"<<endl;
3670                resposta=Menu.menu_sim_nao();
3671                configuracao.opcao_abrir_DT(resposta);///<0 para nao abrir
                        e 1 para abrir
3672                break;
3673
3674              case 4:
3675                resposta=1;
3676                break;
3677
3678
3679              default:
3680                resposta=1;
3681                break;
3682            }
3683
3684            break;
3685
3686
3687          case 3:///<abrir diretorio de trabalho
3688            limpa_janela();
3689
3690            Head.header_menu_abre_DT();
3691            ///verifica sistema operacional
3692            #ifdef __linux
3693            ///sleep(100);
3694            #elif _WIN32
3695            Sleep(100);
3696            #else
3697
3698            #endif
```

```
3699
3700            abre_DT();
3701            resposta=1;
3702
3703            break;
3704
3705
3706          case 4:///abrir diretorio de dados
3707            limpa_janela();
3708
3709            Head.header_menu_abre_bd();
3710            ///verifica sistema operacional
3711            #ifdef __linux
3712            ///sleep(100);
3713            #elif _WIN32
3714            Sleep(100);
3715            #else
3716
3717            #endif
3718
3719            abre_bd();
3720            resposta=1;
3721
3722
3723            break;
3724
3725
3726           case 5:///sair
3727             limpa_janela();
3728             Head.header_titulo();
3729             cout<<"Realmente␣deseja␣sair?"<<endl;
3730             resposta=Menu.menu_sim_nao();
3731             if(resposta==1)
3732               exit(0);
3733             else
3734               resposta=1;
3735             break;
3736
3737
3738           default:
3739
3740             ///verifica sistema operacional
3741             #ifdef __linux
3742             setlocale(LC_ALL, "Portuguese");
3743             cout<<"Op\u00E7\u00E3o␣inv\u00E1lida."<<endl;
3744             locale::global(locale(""));
3745             /// sleep(1000);
3746             #elif _WIN32
```

```
3747            setlocale(LC_ALL, "Portuguese");
3748            cout<<"Opção␣inválida."<<endl;
3749            locale::global(locale(""));
3750            Sleep(1000);
3751            #else
3752
3753            #endif
3754
3755            resposta=1;
3756            break;
3757
3758     }
3759   }
3760   while(resposta==1);
3761 }
3762
3763
3764 void CInterface::abre_DT()
3765 {
3766   ///inicia diretorio de trabalho
3767   ///comando: "move bd\0.txt bd\lixo" Comando para mover o arquivo da
          amostra
3768   ///verifica sistema operacional
3769   programa=configuracao.opcao_programa("explorador");///<define
          explorardor
3770   comando_abrir.str("");
3771   #ifdef __linux
3772   comando_abrir<<programa<<"␣"<<"./DT";
3773   #elif _WIN32
3774   comando_abrir<<"start"<<"␣"<<programa<<"␣"<<"DT";
3775
3776   #else
3777
3778   #endif
3779   system(comando_abrir.str().c_str());///<Manda o comando pro prompt
          para mover o arquivo p lixeira
3780   comando_abrir.str("");
3781 }
3782
3783 void CInterface::abre_relatorio(int int_chave_)
3784 {
3785   ///inicia diretorio de trabalho
3786   ///comando: "move bd\0.txt bd\lixo" Comando para mover o arquivo da
          amostra
3787   ///verifica sistema operacional
3788   programa=configuracao.opcao_programa("explorador");///<define
          explorador
3789   int_chave=int_chave_;
```

```
3790   comando_abrir.str("");
3791   #ifdef __linux
3792   comando_abrir<<programa<<"␣"<<"./DT/relatorios/"<<int_chave;
3793   #elif _WIN32
3794   comando_abrir<<"start"<<"␣"<<programa<<"␣"<<"DT"<<char(92)<<"
          relatorios"<<char(92)<<int_chave;
3795   cout<<comando_abrir.str();
3796   #else
3797
3798   #endif
3799   if(configuracao.opcao_abrir_DT()==1)
3800   {
3801     system(comando_abrir.str().c_str());///<Manda o comando pro prompt
          para mover o arquivo p lixeira
3802   }
3803
3804   comando_abrir.str("");
3805 }
3806
3807 void CInterface::abre_bd()
3808 {
3809   ///inicia diretorio de dados
3810   ///comando: "move bd\0.txt bd\lixo" Comando para mover o arquivo da
          amostra
3811   programa=configuracao.opcao_programa("explorador");//define explorador
3812   ///verifica sistema operacional
3813   comando_abrir.str("");
3814   #ifdef __linux
3815   comando_abrir<<programa<<"␣"<<"./bd";
3816   #elif _WIN32
3817   comando_abrir<<"start"<<"␣"<<programa<<"␣"<<"bd";
3818
3819   #else
3820
3821   #endif
3822   system(comando_abrir.str().c_str());///<Manda o comando pro prompt
          para mover o arquivo p lixeira
3823   comando_abrir.str("");
3824 }
3825
3826 void CInterface::ajusta_janela()
3827 {
3828   ///verifica sistema operacional
3829   #ifdef __linux
3830   ///sleep(100);
3831   #elif _WIN32
3832   ///ajusta cor do programa(comando de MS-DOS)
3833   system("color␣F0");
```

```
3834   ///Maximiza Programa
3835   {///Referencia(http://forum.clubedohardware.com.br/full-screen-c
           /567207 em 15/11/2013)
3836     keybd_event ( VK_MENU, 0x38, 0, 0 );
3837     keybd_event ( VK_SPACE, 0x39, 0, 0 );
3838     keybd_event(0x58,0,0,0);
3839     keybd_event ( VK_MENU, 0x38, KEYEVENTF_KEYUP, 0 );
3840     keybd_event ( VK_SPACE, 0x39, KEYEVENTF_KEYUP, 0 );
3841     keybd_event(0x58,0,KEYEVENTF_KEYUP,0);
3842   }
3843   CHeader Header;
3844   ///Escreve Versao do Windows
3845   cout<<"Versao do sistema: ";
3846   system("VER");
3847   Sleep(100);
3848   #else
3849
3850   #endif
3851
3852 }
3853
3854 void CInterface::limpa_janela()
3855 {
3856   #ifdef __linux
3857   system("clear");///<limpa tela
3858   #elif _WIN32
3859   system("cls");
3860   #else
3861
3862   #endif
3863 }
```

Apresenta-se na listagem 6.11 o arquivo com código da classe CMenu.

Listing 6.11: Arquivo de cabeçalho da classe CMenu.

```
3865 #ifndef CMenu_h
3866 #define CMenu_h
3867
3868 #include <iostream>
3869 #ifdef __linux
3870
3871     #include <cstdlib>
3872
3873 #elif _WIN32
3874         #include <windows.h>///<para configurar a janela
3875         #include <stdlib.h>///<para system
3876 #else
3877
3878 #endif
```

```
3879
3880#include <string>
3881#include <sstream>
3882#include <vector>
3883
3884using namespace std;
3885
3886        ///representa a classe com todos os menus
3887    class CMenu
3888                {
3889                        public:
3890                             int resposta;///<representa a resposta do
                                    usuário
3891                             string resposta_str;///<representa a
                                    resposta do usuário convertida para
                                    string
3892                              char ch;///<representa um caracter
3893                        public:
3894                            int menu_inicio();///<método referente ao
                                   menu de início
3895                            int menu_banco_de_dados();///<método
                                   referente ao menu que lista as funções
                                   referentes ao banco de dados
3896                int menu_exportar_dados();///<menu referente à geração
                       de relatórios
3897                int menu_banco_de_dados_pesquisar();///<menu referente
                       às opções de pesquisa
3898                int menu_filtro();///<menu referente às opções de filtro
3899                int menu_configuracoes();///<menu referente às opções de
                        configurações
3900                        int menu_sim_nao();///<menu referente à
                                   resposta do usuário (sim ou não)
3901                        int menu_exibir_voltar();///<menu referente
                                   às opções exibir ou voltar
3902
3903                };
3904
3905
3906
3907
3908#endif
```

Apresenta-se na listagem 6.12 o arquivo de implementação da classe CMenu.

Listing 6.12: Arquivo de implementação da classe CMenu.

```
3909///verifica sistema operacional
3910        #ifdef __linux
3911        ///#include <unistd.h>
3912        #include <cstdlib>///<para system
```

```
3913         #elif _WIN32
3914
3915         #include <windows.h>
3916         #include <stdlib.h>///<para system
3917         #include <conio.h>
3918
3919         #else
3920
3921         #endif
3922#include <iostream>
3923#include <string>
3924#include <sstream>
3925#include <locale>
3926
3927#include <vector>
3928#include "CMenu.h"
3929#include <stdio.h>
3930
3931
3932
3933using namespace std;
3934
3935int CMenu::menu_inicio()
3936{
3937
3938  #ifdef __linux
3939  cout<<"(1)Banco de Dados"<<endl;
3940  cout<<"(2)Configura\u00E7\u00F5es"<<endl;
3941  cout<<"(3)Abrir Diret\u00F3rio de Trabalho"<<endl;
3942  cout<<"(4)Abrir Diret\u00F3rio de Dados"<<endl;
3943  cout<<"(5)SAIR"<<endl;
3944  cout<<"Informe a Op\u00E7\u00E3o: ";
3945  #elif _WIN32
3946  cout<<"(1)Banco de Dados"<<endl;
3947  cout<<"(2)Configurações"<<endl;
3948  cout<<"(3)Abrir Diretório de Trabalho"<<endl;
3949  cout<<"(4)Abrir Diretório de Dados"<<endl;
3950  cout<<"(5)SAIR"<<endl;
3951  cout<<"Informe a Opção: ";
3952  #else
3953
3954  #endif
3955
3956  cin>>resposta_str;
3957  cin.get();
3958  ///testa se foi digitado um numero maior ou menor
3959  if(atoi(resposta_str.c_str()))
3960  {
```

```
3961     if(atoi(resposta_str.c_str())<1 || atoi(resposta_str.c_str()) >4)
3962     {
3963       resposta=5;///<se resposta for fora do intervalo, define 1 como
             padrao
3964     }
3965     else
3966     {
3967       resposta=atoi(resposta_str.c_str());
3968     }
3969
3970   }
3971   else
3972   {
3973     resposta=5;///<Permanece no MENU
3974   }
3975   return resposta;
3976 }
3977
3978
3979 int CMenu::menu_banco_de_dados()
3980 {
3981   #ifdef __linux
3982
3983   cout<<"(1)Listar"<<endl;
3984   cout<<"(2)Pesquisar"<<endl;
3985   cout<<"(3)Inserir"<<endl;
3986   cout<<"(4)Excluir"<<endl;
3987   cout<<"(5)Exportar"<<endl;
3988   cout<<"(6)VOLTAR"<<endl;
3989   ///cout<<"(8)SAIR"<<endl;
3990   cout<<endl;
3991   cout<<"Infome␣a␣Op\u00E7\u00E3o:␣";
3992
3993   #elif _WIN32
3994
3995   cout<<"(1)Listar"<<endl;
3996   cout<<"(2)Pesquisar"<<endl;
3997   cout<<"(3)Inserir"<<endl;
3998   cout<<"(4)Excluir"<<endl;
3999   cout<<"(5)Exportar"<<endl;
4000   cout<<"(6)VOLTAR"<<endl;
4001   ///cout<<"(8)SAIR"<<endl;
4002   cout<<endl;
4003   cout<<"Infome␣a␣Opção:␣";
4004
4005   #else
4006
4007   #endif
```

```
4008
4009   while(!(cin>>resposta))///<correcao de erro caso entre alguma resposta
           errada
4010   {
4011     cin.clear() ;
4012     cin >> ch;
4013     if(ch==120)break;///<se caracter for x termina processo
4014   }
4015   cin.get();
4016
4017   if(resposta<1 || resposta >7)
4018   {
4019     resposta=7;///<se resposta for fora do intervalo, define 1 como
           padrao
4020   }
4021
4022   return resposta;
4023 }
4024
4025 int CMenu::menu_banco_de_dados_pesquisar()
4026
4027 {
4028   cout<<"(1)Pesquisa␣Simples"<<endl;
4029   cout<<"(2)VOLTAR"<<endl;
4030   cout<<endl;
4031
4032
4033   #ifdef __linux
4034   cout<<"Infome␣a␣Op\u00E7\u00E3o:␣";
4035   #elif _WIN32
4036
4037   cout<<"Infome␣a␣Opção:␣";
4038   #else
4039
4040   #endif
4041
4042   while(!(cin>>resposta))///<correcao de erro caso entre alguma resposta
           errada
4043   {
4044     cin.clear() ;
4045     cin >> ch;
4046     if(ch==120)break;///<se caracter for x termina processo
4047   }
4048   cin.get();
4049
4050   if(resposta<1 || resposta >2)
4051   {
4052     resposta=2;///<se resposta for fora do intervalo, define 1 como
```

```
        padrao
4053    }
4054
4055    return resposta;
4056}
4057
4058int CMenu::menu_configuracoes()
4059{
4060    #ifdef __linux
4061    cout<<"(1)Explorador"<<endl;
4062    cout<<"(2)Editor de Texto"<<endl;
4063    cout<<"(3)Op\u00E7\u00E3o Abrir Diret\u00F3rio de Trabalho (DT)"<<endl
          ;
4064    cout<<endl;
4065    cout<<"(4)Voltar"<<endl;
4066    cout<<endl;
4067    cout<<"Infome a Op\u00E7\u00E3o: ";
4068    #elif _WIN32
4069    cout<<"(1)Explorador"<<endl;
4070    cout<<"(2)Editor de Texto"<<endl;
4071    cout<<"(3)Opção Abrir Diretorio de Trabalho (DT)"<<endl;
4072    cout<<endl;
4073    cout<<"(4)Voltar"<<endl;
4074    cout<<endl;
4075    cout<<"Infome a Opção: ";
4076    #else
4077
4078    #endif
4079
4080    while(!(cin>>resposta))///<correcao de erro caso entre alguma resposta
          errada
4081    {
4082      cin.clear() ;
4083      cin >> ch;
4084      if(ch==120)break;///<se caracter for x termina processo
4085    }
4086    cin.get();
4087    if(resposta<0 || resposta >4)
4088    {
4089      resposta=4;///<se resposta for fora do intervalo, define 1 como
          padrao
4090    }
4091    return resposta;
4092}
4093
4094int CMenu::menu_filtro()
4095{
4096    #ifdef __linux
```

```
4097
4098  cout<<"(-)Chave"<<endl;
4099  cout<<"(2)Nome␣do␣fluido"<<endl;
4100  cout<<"(3)Base"<<endl;
4101  cout<<"(4)pH␣m\u00EDnimo"<<endl;
4102  cout<<"(5)pH␣m\u00C1ximo"<<endl;
4103  cout<<"(6)Teor␣de␣\u00C1gua"<<endl;
4104  cout<<"(7)Peso␣espec\u00EDfico␣m\u00EDnimo"<<endl;
4105  cout<<"(8)Peso␣espec\u00EDfico␣m\u00C1ximo"<<endl;
4106  cout<<"(9)Temperatura␣de␣envelhecimento"<<endl;
4107  cout<<"(10)Salinidade"<<endl;
4108  cout<<"(11)Filtrado"<<endl;
4109  cout<<"|DATA␣DE␣CADASTRO|"<<endl;
4110  cout<<"(12)Dia"<<"␣␣␣␣␣␣␣␣␣␣␣(13)M\u00EAs"<<"␣␣␣␣␣␣␣␣␣(14)Ano"<<endl;
4111  cout<<"(15)Listar␣todos"<<endl;
4112  cout<<"(16)VOLTAR"<<endl;
4113  cout<<endl;
4114  cout<<"Infome␣a␣Op\u00E7\u00E3o:␣";
4115
4116
4117  #elif _WIN32
4118
4119  cout<<"(-)Chave"<<endl;
4120  cout<<"(2)Nome␣do␣fluido"<<endl;
4121  cout<<"(3)Base"<<endl;
4122  cout<<"(4)pH␣m\u00EDnimo"<<endl;
4123  cout<<"(5)pH␣m\u00C1ximo"<<endl;
4124  cout<<"(6)Teor␣de␣\u00C1gua"<<endl;
4125  cout<<"(7)Peso␣espec\u00EDfico␣m\u00EDnimo"<<endl;
4126  cout<<"(8)Peso␣espec\u00EDfico␣m\u00C1ximo"<<endl;
4127  cout<<"(9)Temperatura␣de␣envelhecimento"<<endl;
4128  cout<<"(10)Salinidade"<<endl;
4129  cout<<"(11)Filtrado"<<endl;
4130  cout<<"|DATA␣DE␣CADASTRO|"<<endl;
4131  cout<<"(12)Dia"<<"␣␣␣␣␣␣␣␣␣␣␣(13)M\u00EAs"<<"␣␣␣␣␣␣␣␣␣(14)Ano"<<endl;
4132  cout<<"(15)Listar␣todos"<<endl;
4133  cout<<"(16)VOLTAR"<<endl;
4134  cout<<endl;
4135  cout<<"Infome␣a␣Op\u00E7\u00E3o:␣";
4136
4137  #else
4138
4139  #endif
4140
4141  while(!(cin>>resposta))///<correcao de erro caso entre alguma resposta
          errada
4142  {
4143    cin.clear() ;
```

```
4144      cin >> ch;
4145      if(ch==120)break;///<se caracter for x termina processo
4146    }
4147    cin.get();
4148
4149    if(resposta<1 || resposta >16)
4150    {
4151      resposta=16;
4152    }
4153
4154    return resposta;
4155}
4156
4157
4158int CMenu::menu_sim_nao()
4159{
4160
4161#ifdef __linux
4162
4163      cout<<endl;
4164
4165      setlocale(LC_ALL, "Portuguese");
4166      cout<<"(1)Sim"<<endl;
4167      cout<<"(0)N\u00E3o"<<endl;
4168      cout<<endl;
4169      cout<<"Infome␣a␣Op\u00E7\u00E3o:␣";
4170      locale::global(locale(""));
4171
4172#elif _WIN32
4173      setlocale(LC_ALL, "Portuguese");
4174      cout<<endl;
4175      setlocale(LC_ALL, "Portuguese");
4176      cout<<"(1)Sim"<<endl;
4177      cout<<"(0)Não"<<endl;
4178      cout<<endl;
4179      cout<<"Infome␣a␣Opção:␣";
4180      locale::global(locale(""));
4181
4182#else
4183
4184#endif
4185
4186      while(!(cin>>resposta))///<correcao de erro caso entre alguma
             resposta errada
4187      {
4188        cin.clear() ;
4189        cin >> ch;
4190        if(ch==120)break;///<se caracter for x termina processo
```

```
4191     }
4192     ///<cin.get();
4193 
4194     if(resposta<0 || resposta >1)
4195     {
4196       resposta=0;///<se resposta for fora do intervalo, define 0 como
             padrao
4197     }
4198 
4199     return resposta;
4200 }
4201 
4202 int CMenu::menu_exibir_voltar()
4203 {
4204 
4205   cout<<endl;
4206   cout<<"(0)Voltar"<<endl;
4207   cout<<"(1)Exibir"<<endl;
4208 
4209 
4210   cout<<endl;
4211 
4212   #ifdef __linux
4213   setlocale(LC_ALL, "Portuguese");
4214   cout<<"Infome␣a␣Op\u00E7\u00E3o:␣";
4215   locale::global(locale(""));
4216   #elif _WIN32
4217   setlocale(LC_ALL, "Portuguese");
4218   cout<<"Infome␣a␣Opção:␣";
4219   locale::global(locale(""));
4220   #else
4221 
4222   #endif
4223 
4224 
4225   while(!(cin>>resposta))///<correcao de erro caso entre alguma resposta
         errada
4226   {
4227     cin.clear() ;
4228     cin >> ch;
4229     if(ch==120)break;///<se caracter for x termina processo
4230   }
4231   cin.get();
4232 
4233   if(resposta<0 || resposta >1)
4234   {
4235     resposta=0;///<se resposta for fora do intervalo, define 0 como
           padrao
```

```
4236   }
4237
4238   return resposta;
4239 }
4240
4241 int CMenu::menu_exportar_dados()
4242 {
4243
4244 #ifdef __linux
4245     setlocale(LC_ALL, "Portuguese");
4246     cout<<"(1)Completo"<<endl;
4247     cout<<"(2)VOLTAR"<<endl;
4248     cout<<endl;
4249     cout<<"Infome␣a␣Op\u00E7\u00E3o:␣";
4250
4251     locale::global(locale(""));
4252 #elif _WIN32
4253     setlocale(LC_ALL, "Portuguese");
4254     cout<<"(1)Completo"<<endl;
4255     cout<<"(2)VOLTAR"<<endl;
4256     cout<<endl;
4257     cout<<"Infome␣a␣Opção:␣";
4258
4259     locale::global(locale(""));
4260 #else
4261
4262 #endif
4263
4264     while(!(cin>>resposta))///<correcao de erro caso entre alguma
           resposta errada
4265     {
4266       cin.clear() ;
4267       cin >> ch;
4268       if(ch==120)break;///<se caracter for x termina processo
4269     }
4270     cin.get();
4271     if(resposta<1 || resposta >3)
4272     {
4273       resposta=3;///<se resposta for fora do intervalo, define 3 como
             padrao
4274     }
4275     return resposta;
4276 }
```

Apresenta-se na listagem 6.13 o arquivo com código da classe CRelatorio.

Listing 6.13: Arquivo de cabeçalho da classe CRelatorio.

```
4278 #ifndef CRelatorio_h
4279 #define CRelatorio_h
```

```
4280 #include <string>
4281 #include <vector>
4282 #include <iostream>
4283 #include <fstream>
4284
4285 #include "CFluidodePerfuracao.h"
4286 #include "CBancodeDados.h"
4287
4288 using namespace std;
4289 ///representa a classe que exporta as amostras em forma de relatório
4290 class CRelatorio
4291 {
4292
4293 public:
4294
4295   int chave;///<representa a chave da amostra a ser exportada
4296   int int_chave;///<representa a chave da amostra a ser exportada
4297   int resposta;///<representa a resposta do usuário
4298   unsigned int c;///<contador
4299
4300   vector<CFluidodePerfuracao> v_amostra;///<vetor do tipo vector<
          CFluidodePerfuracao> de amostra
4301
4302   ofstream fout;
4303
4304   stringstream ss;///<conversão em string
4305
4306   CBancodeDados lerbasico;///< cria um objeto da classe CBancodeDados
4307
4308 public:
4309   void exportar_amostra(int);///<método para exportar as amostras em
          forma de relatório
4310 };
4311
4312 #endif
```

Apresenta-se na listagem 6.14 o arquivo de implementação da classe CRelatorio.

Listing 6.14: Arquivo de implementação da classe CRelatorio.

```
4313 #include <iostream>
4314 #include <stdio.h>
4315 #include <fstream>
4316 #include <string>
4317
4318 ///verifica sistema operacional
4319 #ifdef __linux
4320
4321     #include <cstdlib>
4322
```

```
4323 #elif _WIN32
4324
4325     #include <windows.h>
4326     #include <stdlib.h>///<para system
4327     #include <conio.h>
4328
4329 #else
4330
4331 #endif
4332
4333
4334 #include "CRelatorio.h"
4335 #include "CFluidodePerfuracao.h"
4336 #include "CBancodeDados.h"
4337
4338 void CRelatorio::exportar_amostra(int chave_)
4339 {
4340   chave=chave_;
4341
4342   ///criando o diretorio para salvar o relatorio de chave n
4343   #ifdef __linux
4344   ss<<"mkdir␣"<<"DT"<<char(47)<<"relatorios"<<char(47)<<chave;///<DT/
         relatorios/1
4345   #elif _WIN32
4346   ss<<"mkdir␣"<<"DT"<<char(92)<<"relatorios"<<char(92)<<chave;///<DT\
         relatorios\1
4347   #else
4348
4349   #endif
4350
4351   system(ss.str().c_str());///<cria o diretorio a ser gravado os
         arquivos
4352   ss.str("");///<apaga stringstream
4353
4354   ////////////////////////////////////////
4355   /////inicia leitura basica das amostras
4356   ////////////////////////////////////////
4357
4358   ///pesquisa amostra de chave n
4359   int_chave=chave_;
4360   CFluidodePerfuracao amostra_aux;
4361   vector<CFluidodePerfuracao> v_amostra;
4362   unsigned int c;
4363   ///lendo todas as amostras basico
4364   v_amostra=lerbasico.ler_amostra_basico();
4365   ///procurando amostra correspontende a chave informada
4366   for(c=0;c<v_amostra.size();c++)
4367   {
```

```
    if (int_chave==v_amostra[c].chave)
    {
      amostra_aux=v_amostra[c];
      break;
    }
  }
  ////////////////////////////////////////
  /////Termina leitura Basica das amostras
  ////////////////////////////////////////
  ss.str("");///<apaga stringstream

  #ifdef __linux
  ss<<"DT"<<char(47)<<"relatorios"<<char(47)<<chave<<char(47)<<"Amostra-"<<chave<<"-completa.txt";///<
  #elif _WIN32
  ss<<"DT"<<char(92)<<"relatorios"<<char(92)<<chave<<char(92)<<"Amostra-"<<chave<<"-completa.txt";
  #else

  #endif

  fout.open(ss.str().c_str());///<abre arquivo para escrita
  ss.str("");
  //////////////////////////////////////////////////////////////////////////
  ////////////INICIO DA ESCRITA DO CODIGO
  //////////////////////////////////////////////////////////////////////////

  ///inicia das exibicoes basica das amostras
  fout<<"________________________________________________"<<endl;
  fout<<"␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"<<endl;
  fout<<"␣␣␣␣␣␣Relatório␣da␣Amostra␣de␣Chave␣"<<int_chave<<"␣␣␣␣␣"<<endl;
  fout<<"________________________________________________"<<endl;
  fout<<endl<<endl;
  fout<<"Data␣de␣Cadastro␣----␣:␣"<<amostra_aux.dia<<"/"<<amostra_aux.mes<<"/"<<amostra_aux.ano<<endl;
  fout<<"Nome␣do␣Fluido␣-------:␣"<<amostra_aux.nome_do_fluido<<endl;
  fout<<"Base␣---------------␣:␣"<<amostra_aux.base<<endl;
  fout<<"Teor␣de␣Base␣--------␣:␣"<<lerbasico.teste_vazio(amostra_aux.teorbase)<<endl;
  fout<<"Teor␣de␣Agua␣--------␣:␣"<<lerbasico.teste_vazio(amostra_aux.teoragua)<<endl;
  fout<<"␣␣␣␣-----Propriedades␣Físicas␣e␣químicas-----␣␣␣␣␣␣"<<endl;
  fout<<"pH␣Mínimo␣-----------␣:␣"<<lerbasico.teste_vazio(amostra_aux.
```

```
        pH_min)<<endl;
4406    fout<<"pH Máximo ---------- : "<<lerbasico.teste_vazio(amostra_aux.
        pH_max)<<endl;
4407    fout<<"Peso Específico Mínimo: "<<lerbasico.teste_vazio(amostra_aux.
        pe_min)<<endl;
4408    fout<<"Peso Específico Máximo: "<<lerbasico.teste_vazio(amostra_aux.
        pe_max)<<endl;
4409    fout<<"Temperatura de envelhecimento: "<<lerbasico.teste_vazio(
        amostra_aux.temp_e)<<endl;
4410    fout<<"Força gel inicial antes do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.gel_i_ae)<<endl;
4411    fout<<"Força gel inicial depois do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.gel_i_de)<<endl;
4412    fout<<"Força gel final antes do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.gel_f_ae)<<endl;
4413    fout<<"Força gel final depois do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.gel_f_de)<<endl;
4414    fout<<"Estabilidade elétrica antes do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.est_el_ae)<<endl;
4415    fout<<"Estabilidade elétrica depois do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.est_el_de)<<endl;
4416    fout<<"Coeficiente de lubricidade: "<<lerbasico.teste_vazio(
        amostra_aux.c_lubricidade)<<endl;
4417    fout<<"Volume de filtrado: "<<lerbasico.teste_vazio(amostra_aux.
        filtrado)<<endl;
4418    fout<<"Teor de sólidos ----- : "<<lerbasico.teste_vazio(amostra_aux.
        teor_solidos)<<endl;
4419    fout<<"Salinidade ---------- : "<<lerbasico.teste_vazio(amostra_aux.
        temp_e)<<endl;
4420    fout<<"     ----Propriedades Reológicas ----      "<<endl;
4421    fout<<"Viscosidade Aparente antes do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.VA_ae)<<endl;
4422    fout<<"Viscosidade Aparente depois do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.VA_de)<<endl;
4423    fout<<"Viscosidade Plástica antes do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.VP_ae)<<endl;
4424    fout<<"Viscosidade Plástica depois do envelhecimento: "<<lerbasico.
        teste_vazio(amostra_aux.VP_de)<<endl;
4425    fout<<"     ----Aditivos ----      "<<endl;
4426    fout<<"Adensante ---------- : "<<amostra_aux.adensante<<endl;
4427    fout<<"Concentração de Adensante ---------- : "<<lerbasico.teste_vazio
        (amostra_aux.conc_adensante)<<endl;
4428    fout<<"Inibidor de Formações Ativas ------- : "<<amostra_aux.
        inibidor_fa<<endl;
4429    fout<<"Concentração do Inibidor de Formações Ativas ---------- : "<<
        lerbasico.teste_vazio(amostra_aux.conc_inibidor_fa)<<endl;
4430    fout<<"Redutor de Filtrado - : "<<amostra_aux.redutor_f<<endl;
4431    fout<<"Concentração do Redutor de Filtrado ---------- : "<<lerbasico.
```

```
        teste_vazio(amostra_aux.conc_redutor_f)<<endl;
4432    fout<<"Biopolimero - :"<<amostra_aux.biopolimero<<endl;
4433    fout<<"Concentração do Biopolimero ---------- :"<<lerbasico.
        teste_vazio(amostra_aux.conc_biopolimero)<<endl;
4434    fout<<"Viscosificante - :"<<amostra_aux.viscosificante<<endl;
4435    fout<<"Concentração do Viscosificante ---------- :"<<lerbasico.
        teste_vazio(amostra_aux.conc_viscosificante)<<endl;
4436    fout<<"Dispersante - :"<<amostra_aux.dispersante<<endl;
4437    fout<<"Concentração do Dispersante ---------- :"<<lerbasico.
        teste_vazio(amostra_aux.conc_dispersante)<<endl;
4438    fout<<"Defloculante - :"<<amostra_aux.defloculante<<endl;
4439    fout<<"Concentração do Defloculante ---------- :"<<lerbasico.
        teste_vazio(amostra_aux.conc_defloculante)<<endl;
4440    fout<<"Emulsificante - :"<<amostra_aux.emulsificante<<endl;
4441    fout<<"Concentração do Emulsificante ---------- :"<<lerbasico.
        teste_vazio(amostra_aux.conc_emulsificante)<<endl;
4442    fout<<"Biocida - :"<<amostra_aux.biocida<<endl;
4443    fout<<"Concentração do Biocida ---------- :"<<lerbasico.teste_vazio(
        amostra_aux.conc_biocida)<<endl;
4444    fout<<"Lubrificante - :"<<amostra_aux.lubrificante<<endl;
4445    fout<<"Concentração do Lubrificante ---------- :"<<lerbasico.
        teste_vazio(amostra_aux.conc_lubrificante)<<endl;
4446    fout<<"Inibidor de corrosão - :"<<amostra_aux.inibidor_c<<endl;
4447    fout<<"Concentração do Inibidor de Corrosão ---------- :"<<lerbasico.
        teste_vazio(amostra_aux.conc_inibidor_c)<<endl;
4448    fout<<"Controlador de pH - :"<<amostra_aux.controlador_pH<<endl;
4449    fout<<"Concentração do Controlador de pH ---------- :"<<lerbasico.
        teste_vazio(amostra_aux.conc_controlador_pH)<<endl;
4450    fout<<endl<<endl;
4451
4452
4453    ////////////////////////////////////////////////////////
4454    ////////FIM DA ESCRITA DO CODIGO
4455    ////////////////////////////////////////////////////////
4456
4457    fout.close();
4458}
```

Apresentam-se na listagem **??** o programa que usa as classes.

Listing 6.15: Arquivo de implementação da função `main()`.

```
4463#include <iostream>
4464
4465    #ifdef __linux__
4466        #elif _WIN32
4467#include <windows.h>
4468#include <stdlib.h>///<para system
4469        #else
4470        #endif
```

```
4471
4472#include "CInterface.h"
4473
4474using namespace std;
4475
4476///int main(int argc, char** argv)
4477int main()
4478{
4479
4480         ///Inicia a interface do programa
4481         CInterface Interface;
4482         Interface.inicia();
4483return(0);
4484}
```

Capítulo 7

Teste

Todo projeto de engenharia passa por uma etapa de testes. Neste capítulo apresentamos alguns testes do software desenvolvido.

7.1 Teste 1: Acesso ao Banco de Dados

No seguinte teste apresentado na imagem mostraremos o menu inicial, onde o usuário pode ter acesso ao Banco de Dados e às funções de Configurações, Abrir Diretório de Trabalho e Abrir Diretório de Dados através de uma interface em modo texto.

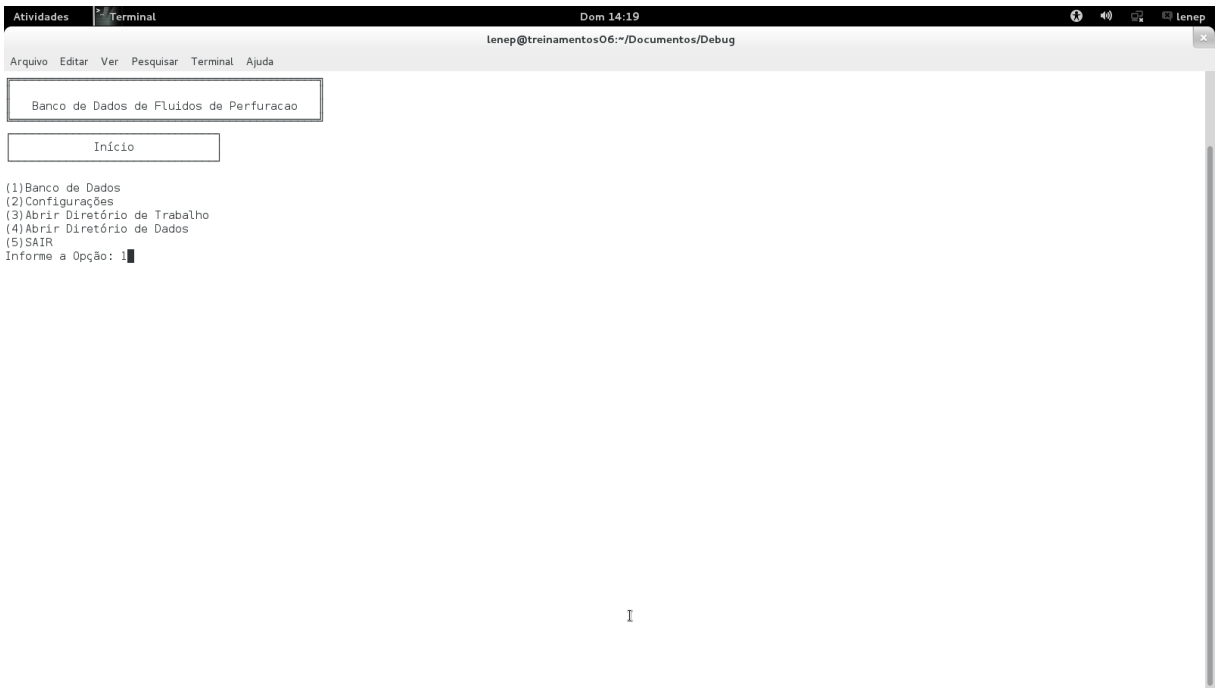


Figura 7.1: Tela do programa mostrando o menu inicial

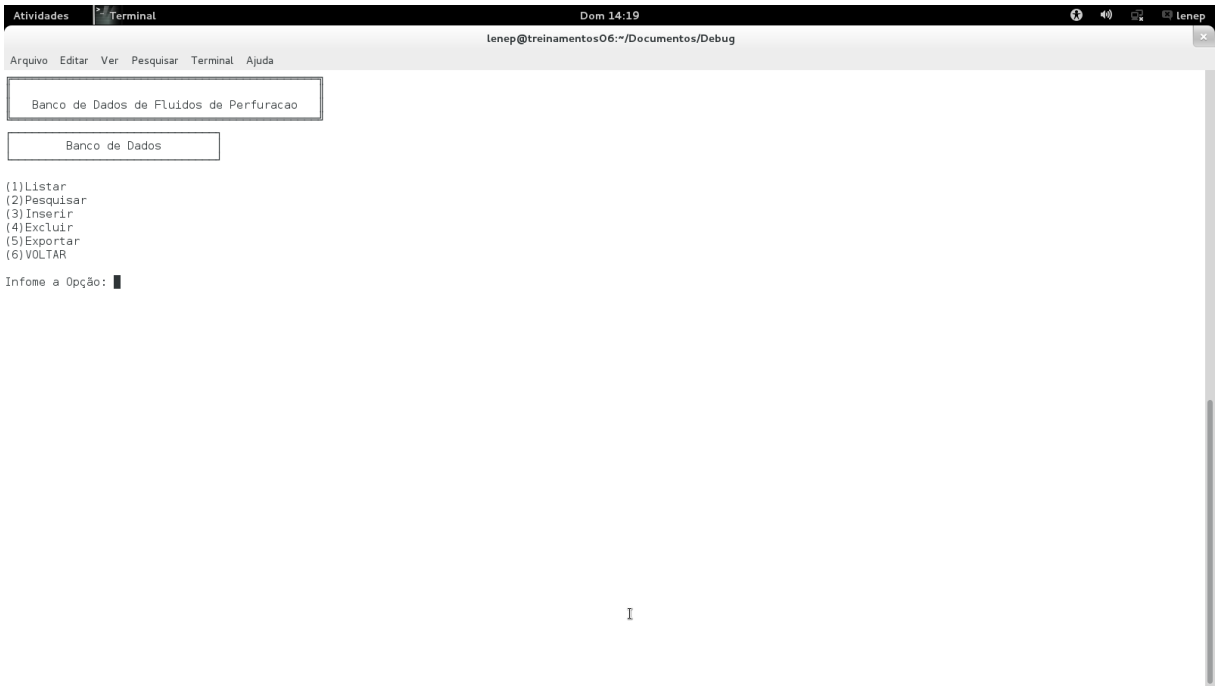


Figura 7.2: Tela do programa mostrando o menu banco de dados

7.2 Teste 2: Listar Fluidos de Perfuração

Após acessar o Banco de Dados, o usuário terá acesso ao menu com diferentes opções. Após escolher listar os fluidos, o programa apresentará a seguinte tela mostrada na imagem abaixo.

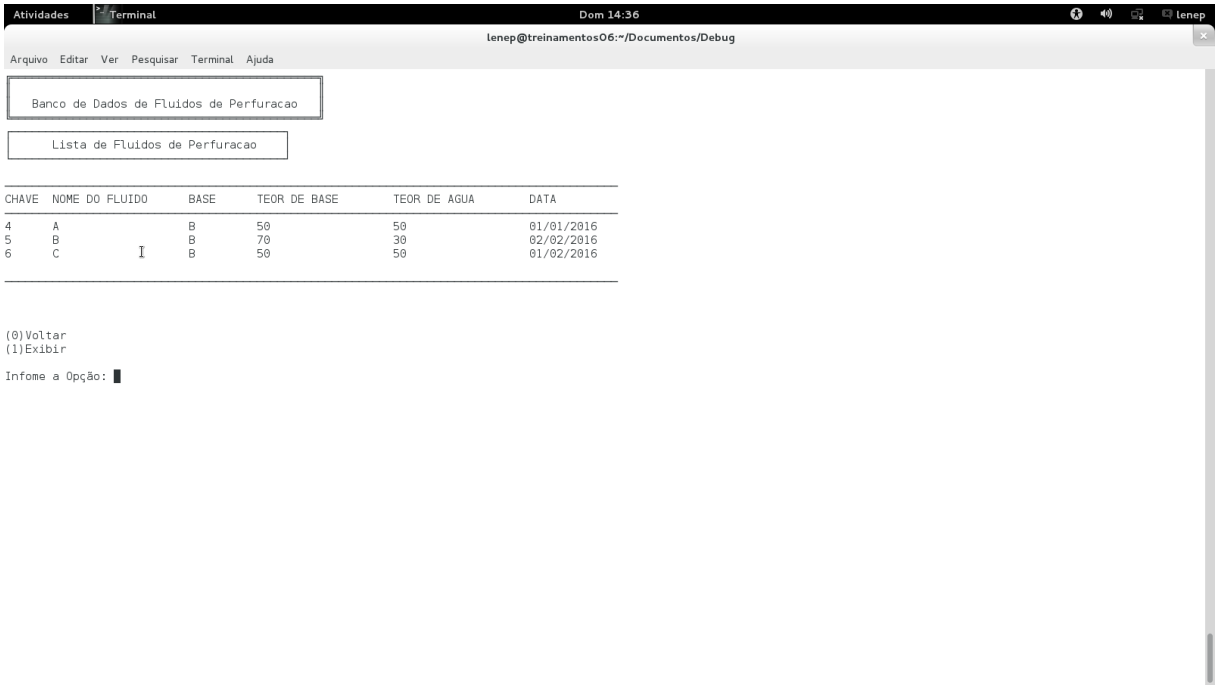


Figura 7.3: Tela do programa mostrando a lista de fluidos de perfuração

Após o programa ter exibido as amostras selecionadas, aparecerá para o usuário a

opção de exibir alguma amostra. Abaixo apresenta-se uma amostra exibida.

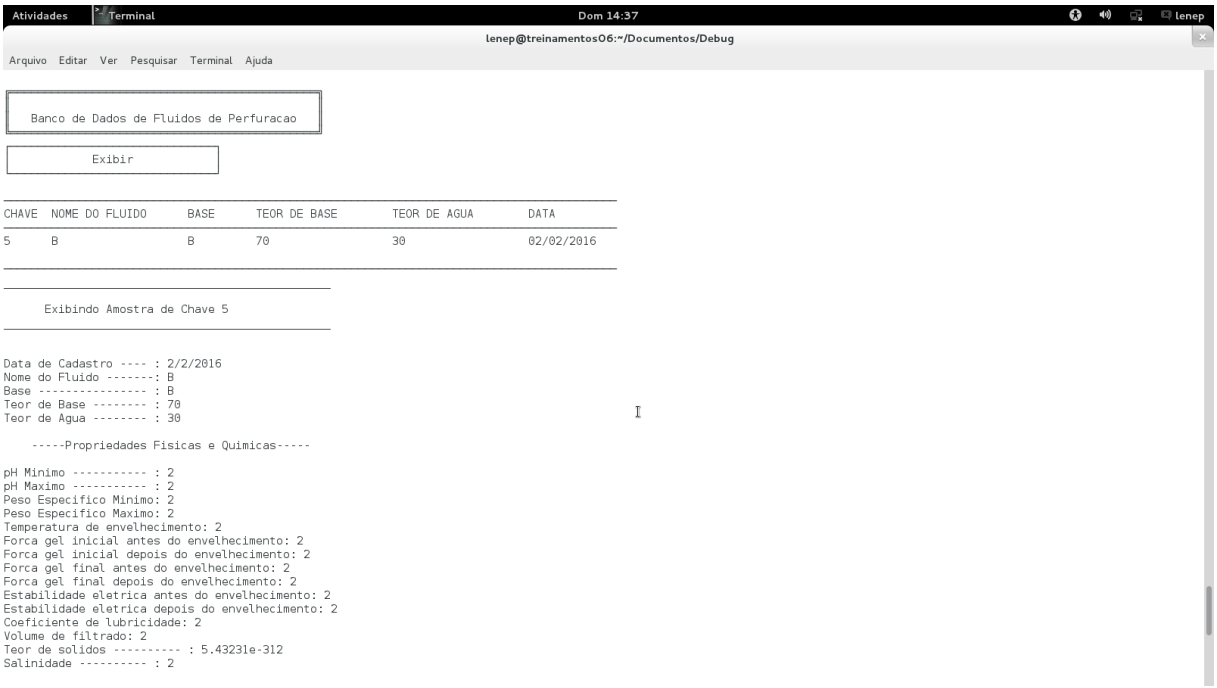


Figura 7.4: Tela do programa mostrando um fluido de perfuração sendo exibido

7.3 Teste 3: Pesquisar Fluidos de Perfuração

Após acessar o Banco de Dados, o usuário terá acesso ao menu com diferentes opções. Após escolher pesquisar os fluidos, o programa apresentará a seguinte tela mostrada na imagem abaixo. O usuário poderá escolher pesquisar os fluidos por diferentes tipos de filtros apresentados na captura de tela abaixo.

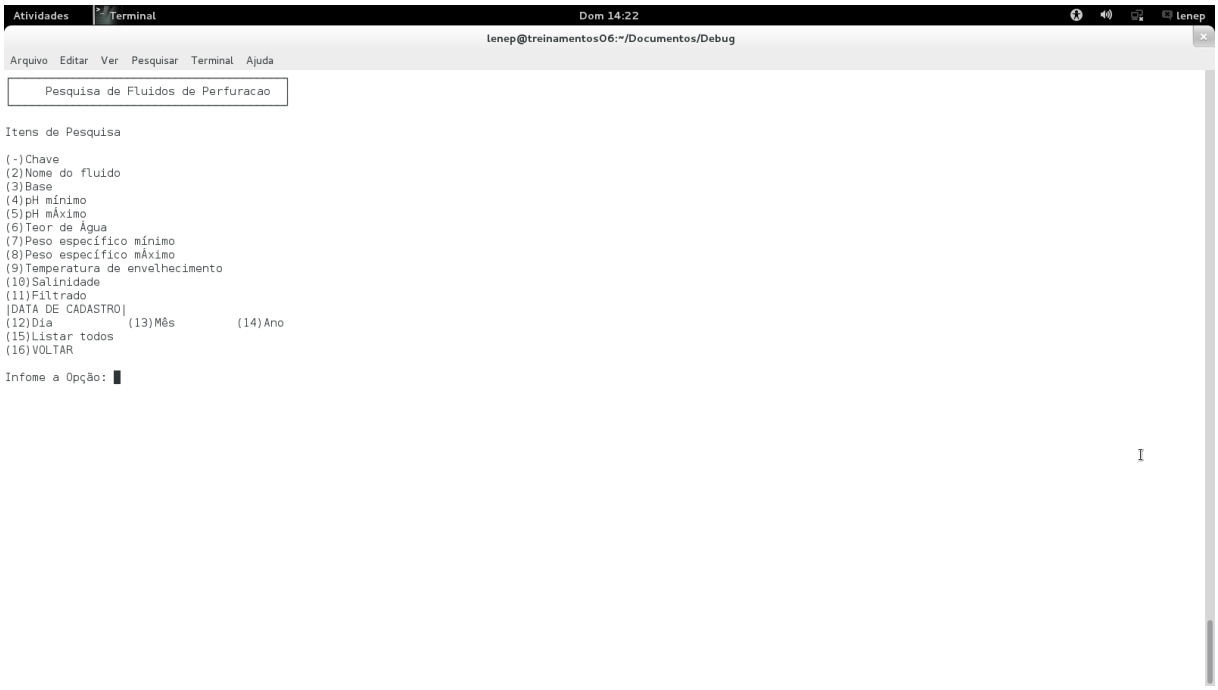


Figura 7.5: Tela do programa mostrando os filtros que podem ser utilizados para a pesquisa de fluidos de perfuração

Depois de escolher o filtro, o usuário terá como resposta os fluidos pesquisados com o filtro selecionado. Segue um exemplo de filtro na imagem abaixo.

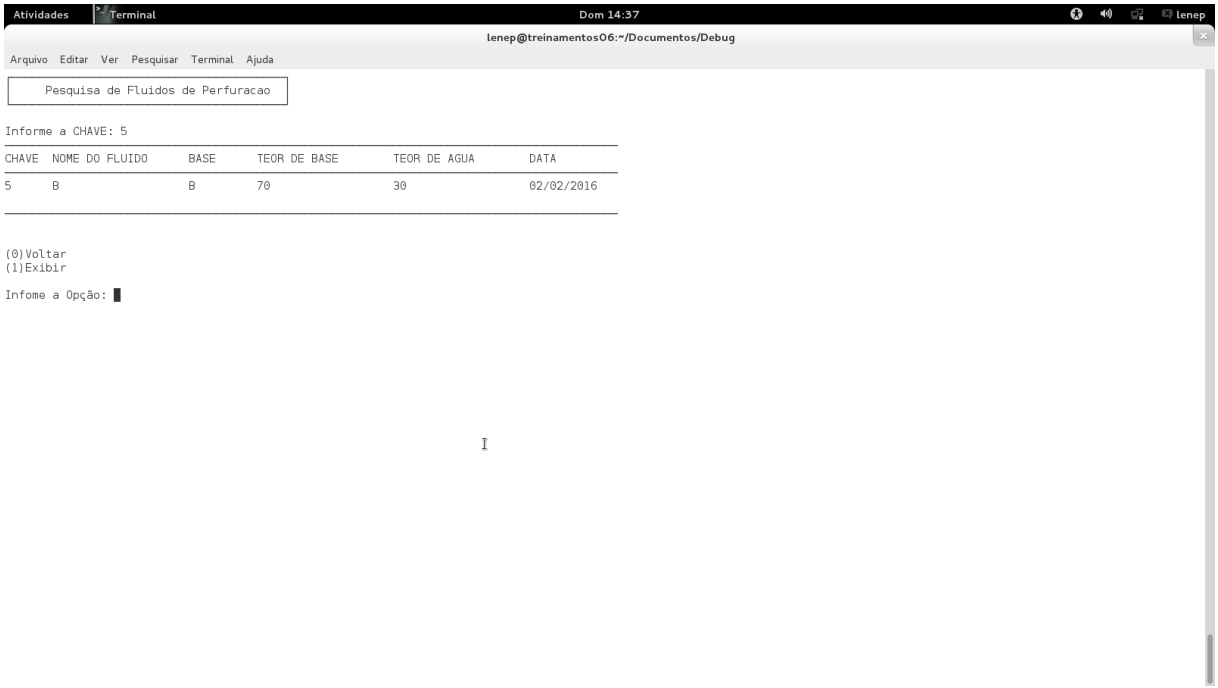


Figura 7.6: Tela do programa mostrando a pesquisa de fluidos de perfuração

7.4 Teste 4: Inserir Fluidos de Perfuração

Após acessar o Banco de Dados, o usuário terá acesso ao menu com diferentes opções. Após escolher inserir fluidos, o usuário deverá ir caracterizando o fluido a ser inserido com alguns de seus dados, como os apresentados na imagem abaixo.

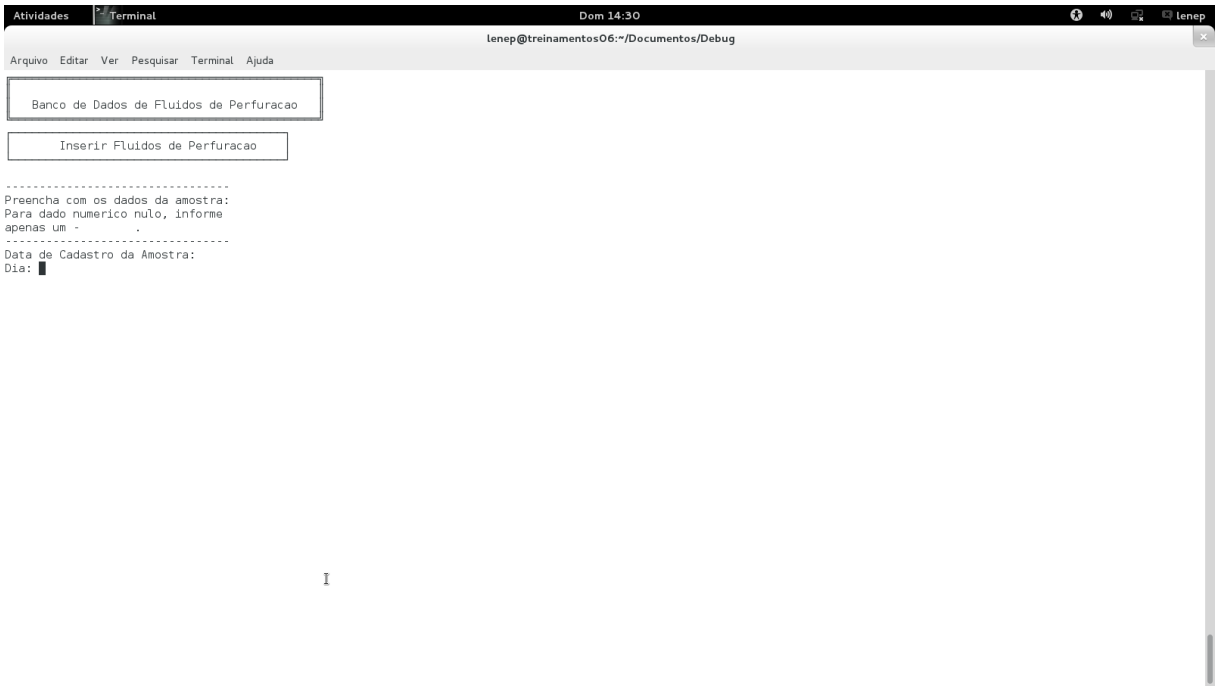


Figura 7.7: Tela do programa mostrando alguns dados a serem respondidos pelo usuário na inserção de fluidos de perfuração

7.5 Teste 5: Exportar Fluidos de Perfuração

Após acessar o Banco de Dados, o usuário terá acesso ao menu com diferentes opções. Após escolher exportar fluidos, o usuário terá a opção de exportar a amostra de forma completa ou de voltar. Escolhendo a opção de exportar de forma completa, o usuário deverá escolher a chave a ser exportada e depois ele terá como resposta um arquivo no formato txt na pasta relatorios. A imagem abaixo mostra uma captura de tela de um relatório gerado.

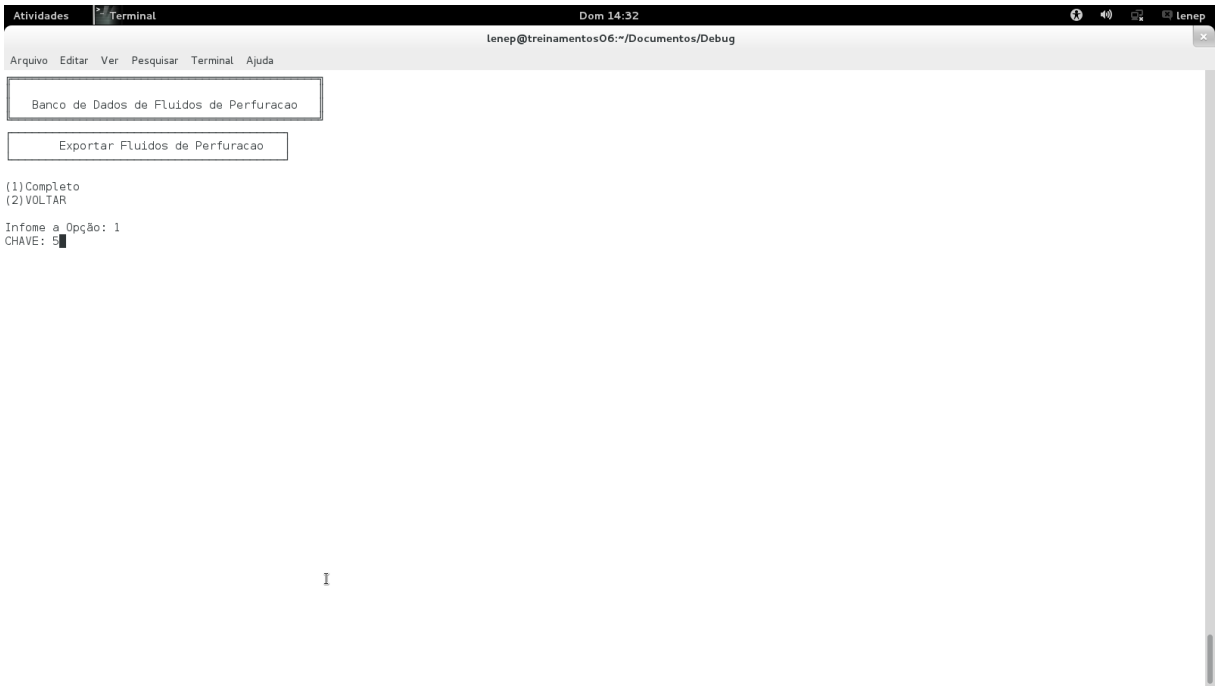


Figura 7.8: Tela do programa mostrando o menu para exportar fluidos de perfuração

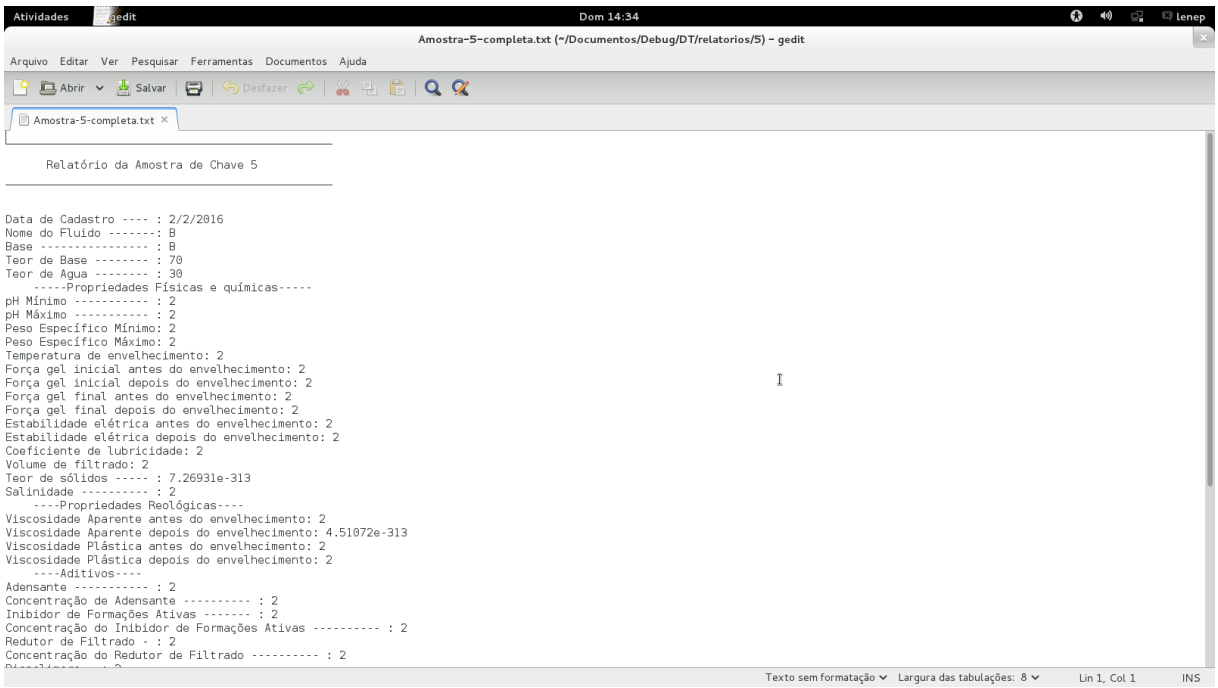


Figura 7.9: Tela do programa mostrando o relatório gerado de um fluido de perfuração

7.6 Teste 6: Excluir Fluidos de Perfuração

Após acessar o Banco de Dados, o usuário terá acesso ao menu com diferentes opções. Após escolher excluir fluidos, o usuário deverá escolher a chave a ser excluída. A imagem abaixo mostra uma captura de tela do menu de exclusão do fluido.

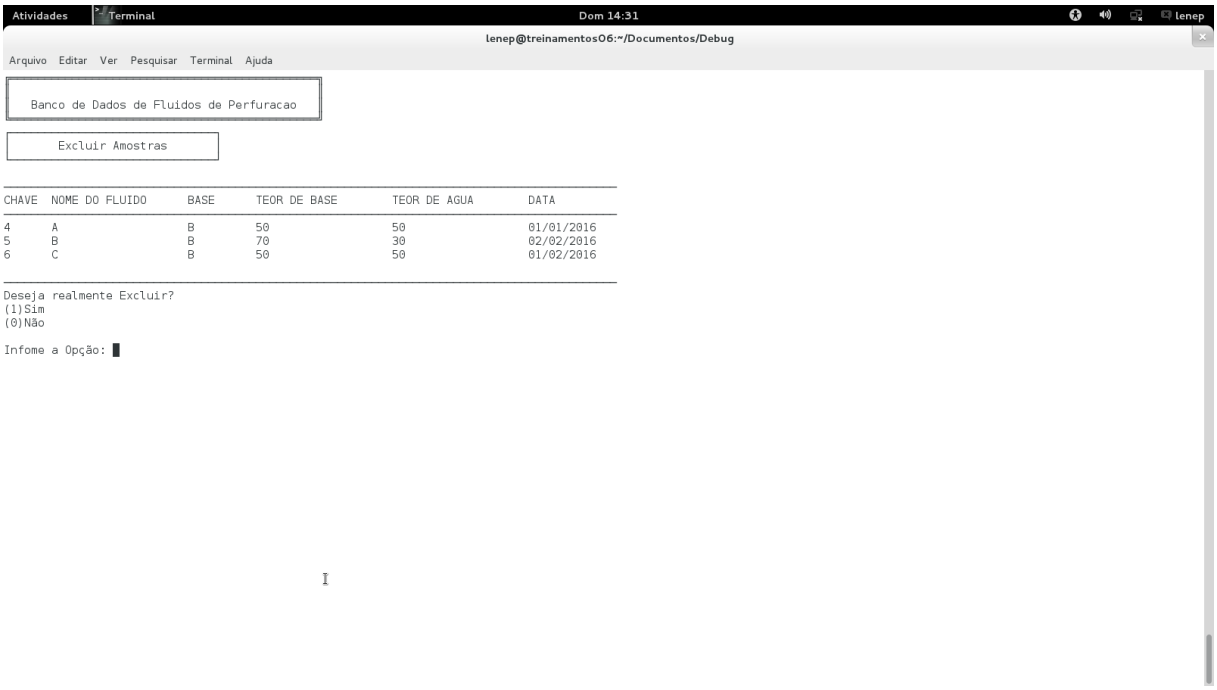


Figura 7.10: Tela do programa mostrando o menu de exclusão de fluidos de perfuração

Capítulo 8

Documentação

A presente documentação refere-se ao uso do Banco de Dados de Fluidos de Perfuração.

8.1 Documentação do Usuário

Abaixo encontra-se uma explicação de como o usuário pode utilizar o programa.

8.1.1 Como utilizar o software

Após abrir o terminal, compilar o programa e, depois, executá-lo, o usuário terá um menu inicial com as seguintes opções:

1. Banco de Dados
2. Configurações
3. Abrir Diretório de Trabalho
4. Abrir Diretório de Dados

Se o usuário escolher a opção Banco de Dados, um menu referente ao Banco de Dados será apresentado, com as seguintes opções:

1. Listar
2. Pesquisar
3. Inserir
4. Excluir
5. Exportar
6. Voltar

Uma vez escolhida a opção de Listar, o usuário terá os fluidos de perfuração presentes no Banco de Dados listados com algumas características principais, além da opção de poder exibir fluidos selecionados.

Se a opção escolhida for a de pesquisar, o usuário poderá pesquisar fluidos com os seguintes filtros:

1. Chave
2. Nome do fluido
3. Base
4. pH mínimo
5. pH máximo
6. Teor de água
7. Peso específico mínimo
8. Peso específico máximo
9. Temperatura de envelhecimento
10. Salinidade
11. Filtrado
12. Dia
13. Mês
14. Ano
15. Listar todos
16. VOLTAR

Se o usuário selecionar a opção Inserir, ele deverá fornecer alguns dados para a caracterização dos diferentes fluidos.

Escolhendo a opção excluir, o usuário deverá selecionar a chave do fluido de perfuração a ser excluída.

Por fim, ao selecionar a opção exportar, o usuário terá um relatório do fluido selecionado na forma de um arquivo no formato txt.

Ao escolher a opção Configurações no menu inicial, o usuário poderá configurar o programa explorador e o editor de textos.

Na Opção Abir Diretório de Trabalho e Abrir Diretório de Dados, o usuário poderá abrir os respectivos diretórios.

8.2 Documentação para Desenvolvedor

A documentação abaixo é apresentada para usuários que queiram modificar, aperfeiçoar ou ampliar este software.

8.2.1 Dependências

Para compilar o software é necessário atender as seguintes dependências:

- No sistema operacional GNU/Linux:
  - Instalar o compilador g++ da GNU disponível em http://gcc.gnu.org.
  - Para instalar no GNU/Linux use o comando yum install gcc.
- No sistema operacional Windows:
  - Instalar um compilador apropriado.
  - Recomenda-se o Dev C++ disponível em http://dev-c.softonic.com.br/.

8.2.2 Documentação usando doxygen

. O software `doxygen` lê os arquivos com os códigos (*.h e *.cpp) e gera uma documentação muito útil e de fácil navegação no formato html. Abaixo encontram-se algumas imagens referentes à documentação usando doxygen.

8.3 Referências Bibliográficas

- CUNHA, R. R. Estudo preliminar sobre o uso de glicerina proveniente da produção de biodiesel como base para fluidos de perfuração dos poços de petróleo. Macaé/ RJ. 2015.

Índice Remissivo