Rodolfo Wilvert Reitz

DESENVOLVIMENTO DE UMA FERRAMENTA PARA ANÁLISE E TRATAMENTO DE DADOS

Trabalho de conclusão de curso submetido ao Curso de Bacharelado em Ciências da Computação para a obtenção do Grau de Bacharel em Ciências da Computação.
Orientador: Prof. Dr. Paulo José de Freitas Filho

Florianópolis

2014

Ficha de identificação da obra elaborada pelo autor através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

A ficha de identificação é elaborada pelo próprio autor

Maiores informações em:
http://portalbu.ufsc.br/ficha

Rodolfo Wilvert Reitz

DESENVOLVIMENTO DE UMA FERRAMENTA PARA ANÁLISE E TRATAMENTO DE DADOS

Este Trabalho de conclusão de curso foi julgado aprovado para a obtenção do Título de "Bacharel em Ciências da Computação", e aprovado em sua forma final pelo Curso de Bacharelado em Ciências da Computação.

Florianópolis, 3 de dezembro 2014.

Prof. Dr. Renato Cislaghi
Coordenador

Banca Examinadora:

Prof. Dr. Paulo José de Freitas Filho
Orientador

Prof. Dr. Mauro Roisenberg

Prof. Dr. Sílvia Modesto Nassar

Este trabalho é dedicado aos meus queridos pais Francisco e Arlete, e minha noiva Lídia que sempre me incentivaram a seguir em frente, ajudando a tornar esta conquista possível.

AGRADECIMENTOS

A Deus por ter me dado esta oportunidade, saúde e força para superar as dificuldades.

Ao meu orientador, professor Paulo, pelo suporte no tempo que lhe coube, pelas suas correções e incentivos.

Aos meus pais, pelo amor, incentivo e apoio incondicional.

Aos membros do laboratório que direta ou indiretamente fizeram parte da minha formação.

A Petrobras pela bolsa recebida durante o desenvolvimento deste trabalho.

A todos estes, o meu muito obrigado.

RESUMO

A análise exploratória dos dados (AED) consiste em utilizar ferramentas gráficas e descritivas para analisar um conjunto de dados afim de extrair informações sobre ele. O tratamento dos dados (TD) é uma etapa complementar a AED e consiste de estratégias afim de melhorar ou adequar o conjunto de dados.
Neste trabalho é realizado um levantamento de métodos de AED e TD, que em seguida foram reunidos em uma ferramenta desenvolvida em C++ e utilizando Qt Framework para desenhar a interface gráfica.
A ferramenta desenvolvida contém em termos de AED: medidas descritivas, distribuições de frequências, gráficos, detecção de valores anormais (*outliers*, inválidos e *missings*). Enquanto em termos de TD, a ferramenta desenvolvida possui funcionalidades que auxiliam a adequação dos dados, como preenchimento/estimação de valores utilizando média, regressões lineares simples ou distribuições de probabilidades.
Esta ferramenta foi desenvolvida com a finalidade de ser integrada a um dos projetos do qual o autor deste trabalho participa.

Palavras-chave: análise, tratamento, dados, c++, qt

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

LISTA DE SÍMBOLOS

LISTA DE LISTAGENS

SUMÁRIO

1 INTRODUÇÃO

1.1 INTRODUÇÃO GERAL

Segundo Gantz e Reinsel (2007), estima-se que a quantidade total de informações digitais presente no universo entre os anos de 2006 a 2010 aumentaria de 161 para 988 *exabytes*, um crescimento esperado anualmente de 57%. Este massivo incremento é um dos principais aspectos tratados pelo Big Data e sua causa está fortemente relacionada com a popularização de equipamentos de captura imagem, vídeo e áudio. Entretanto, uma outro motivo se deve, segundo Lohr (2012), ao fato que atualmente a maioria dos sistemas, como as cidades inteligentes ou redes de computadores, têm consigo sensores que monitoram ao longo do tempo variáveis referentes ao sistema em questão, de modo a comunicar as medições capturadas para algo que irá utilizá-la para tomar decisões necessárias para que o sistema alcance seu objetivo.

Tendo em vista esta crescente quantidade de informações, é cada vez maior a probabilidade da ocorrência de erros durante a sua captura, promovendo a redução da qualidade dos dados, de modo que eles não representem corretamente a realidade. Segundo Silva (2011), os erros mais comuns são:

- De medições.
- De transcrição.
- Diferentes formatos de representação.
- Heterogeneidade, por exemplo, quando múltiplos sensores desigualmente calibrados monitoram uma mesma variável.
- Por algum motivo o valor a ser medido não estava disponível no momento.

Uma forma de tentar encontrar estes problemas é realizando uma AED (Análise exploratória dos dados), que consiste, segundo Reis (2008), na coleta, apresentação, análise e interpretação dos dados através de instrumentos adequados: tabelas, gráficos e indicadores. Já o TD (Tratamento dos dados) consiste em adequar os problemas identificados de modo a melhor condizer com o esperado. Ambas, AED e TD juntas podem ser chamadas de ATD (Análise e tratamento dos dados).

A ATD é de grande importância para as mais diversas aplicações que são baseadas em dados de entradas, dois exemplos serão listados a seguir:

- Aprendizado de máquina: Área da inteligência artificial dedicada na construção de modelos computacionais que possibilitam o computador aprender (extrair regras e padrões) a partir dos dados e utilizá-lo, por exemplo, para realizar previsões.

- Simulação de sistemas: Técnica utilizada com objetivo de imitar ou simular as operações de um processo em questão (sistema). A maioria dos sistemas possuem comportamento estocástico, afim de fazer com que a simulação possua o mesmo comportamento são utilizadas, por exemplo, distribuições de probabilidades para representar cada variável aleatória do sistema real, sendo que estas distribuições têm seus parâmetros obtidos através especialistas ou estimadores, contudo ambos se apoiam em dados do sistema real.

Essas são duas de muitas aplicação onde a ATD é essencial para o seu funcionamento, ambas são fortemente dirigidas por dados, de modo que a qualidade dos resultados estão diretamente relacionados com a qualidade da entrada. A qualidade da entrada determina a confiança de utilizar estes sistemas para apoio à tomadas de decisões.

1.2 MOTIVAÇÃO

Modelos de previsão são essenciais para qualquer sistema (transporte, produção, administrativo, etc.), pois conseguem prever os efeitos da execução de um plano e ajudam nas tomadas de decisões relacionas com questões do tipo "o que aconteceria se?". Entretanto, a existência de dados que não condizem com a realidade podem comprometer o desempenho destes modelos. Então, se faz necessário a ATD para buscar e corrigir o maior número possível de problemas afim de aumentar a qualidade dos dados e consequentemente elevar a confiança destes modelos.

A principal motivação surgiu de um dos projetos que o autor deste trabalho participa durante a graduação junto ao Performance-Lab, na qual utiliza modelos computacionais para prever e otimizar o processo de perfuração de petróleo, titulado SO-BR (Sistema de Otimização de Brocas). Afim de buscar melhores resultados e dispor de algo para visualizar e editar os dados de entrada, surgiu a oportunidade

de integrar ao projeto um módulo de ATD, esse módulo é a ferramenta desenvolvida neste trabalho.

No início, o orientador e os membros da banca deste trabalho forneceram o código fonte de um projeto já desenvolvido pelo PerformanceLab, chamado de E&P Risk, na qual contém um módulo de ATD que foi codificado por Santin (2007) em outro framework (C++ Builder) diferente do projeto SO-BR (Qt Framework). Este módulo de ATD do E&P Risk foi utilizado como apoio, porém foi amplamente melhorado em termos de codificação (*bugs*, otimizações), assim como novas funcionalidades foram adicionadas.

Uma outra motivação é que a ferramenta desenvolvida é de uso geral para qualquer área, podendo ser facilmente integrada a qualquer outro projeto, desde que utilize o mesmo framework (Qt) e a mesma linguagem (C++), também por ser de código aberto, está disponível para toda a sociedade acadêmica para utilização, estudo ou aperfeiçoamento.

1.3 OBJETIVO GERAL

Desenvolvimento de uma ferramenta que forneça suporte a ATD, que posteriormente foi integrada como um módulo no SO-BR.

1.4 OBJETIVOS ESPECÍFICOS

- Migrar e aperfeiçoar o módulo de ATD do E&P Risk desenvolvido em C++ Builder para Qt Framework.
- Investigar os métodos de AED: medidas descritivas e representações gráficas.
- Investigar estratégia de apoio ao TD, tais como preencher o valor inadequado utilizando distribuições de probabilidade ou regressões lineares.
- Codificar os conceitos estudados e desenvolver uma interface gráfica que permita utilizar as técnicas de ATD estudadas.

1.5 ESTRUTURA DO TRABALHO

A estrutura do trabalho se dará da forma apresentada na sequência:

No presente capítulo, apresenta-se a introdução que é composta pela introdução geral, motivação, objetivos gerais e específicos e a estrutura do trabalho.

O segundo capítulo expõe conceitos de ATD utilizadas para criar a ferramenta.

No terceiro capítulo, é apresentado o desenvolvimento da ferramenta: metodologia, organização, funcionamento, aspectos de implementação e a interface gráfica desenvolvida.

No quarto capítulo e último é apresenta uma conclusão contendo: lista com as principais alterações em relação ao módulo de ATD do E&P Risk, citação de algumas outras ferramentas existentes, uma breve discussão do trabalho desenvolvido e sugestões de trabalhos futuros.

2 CONCEITOS FUNDAMENTAIS

Este capítulo descreve os conceitos utilizados para criar a ferramenta.

2.1 ANÁLISE EXPLORATÓRIA DOS DADOS

A AED ou estatística descritiva consiste, segundo MEDRI (2011), de um conjunto estratégias de organização, apresentação e sintetização dos dados, utilizando gráficos, tabelas e medidas descritivas para estudar um conjunto de dados, de modo a transformar os dados brutos em informações. Segundo Todesco (2014) a AED é utilizada, por exemplo, como pré-requisito para uma análise dos dados mais formal (Predição, Previsão, Estimação, Classificação e Testes de Hipóteses), e como parte integral formal da construção de modelos, pois dá suporte à uma melhor compreensão dos dados e eventual detecção problemas que podem diminuir o desempenho ou a confiança destes modelos.

2.1.1 Dados e atributos

Os dados são geralmente organizados em forma de tabela, onde cada linha corresponde a uma observação, indivíduo ou registro, cada coluna corresponde a uma característica, variável ou atributo e cada célula é um valor do registro (BARBETTA; REIS; BORNIA, 2010). Neste trabalho foram utilizados os termos atributo, registro e valor .

Os atributos se classificam entre quantitativos (números de certa escala) ou qualitativos (qualidade ou categoria). A Tabela 1 é um exemplo de dados climáticos provenientes do BDMEP (Banco de Dados Meteorológicos para Ensino e Pesquisa). Os atributos NumDiasPrecipitação e PrecipitaçãoTotal são considerados quantitativos, enquanto o atributo Estação é considerado qualitativo. Os atributos quantitativos ainda se dividem em dois subconjuntos: discretos e contínuos. Pela Tabela 1, o atributo Mês é um exemplo de atributo quantitativo discreto, pois possui valores que podem ser listados (1, 2, 3, ..., 12), caso o atributo Mês fosse representado textualmente ele seria qualitativo. Enquanto PrecipitaçãoTotal, que pode assumir qualquer valor é um atributo quantitativo contínuo. Os atributos qualitativos também são classificados em ordinal e nominal, que significam respectivamente

a existência e inexistência de uma ordenação das categorias do atributo, porém estas duas subclassificações não foram englobadas neste trabalho.

Tabela 1: Exemplo de dados

id	Estação	Mês	NumDiasPrecipitação	PrecipitaçãoTotal
0	RS	12	9	134,1
1	RS	12	5	48,2
2	SC	12	20	138,1
3	SC	12	13	35,6
4	SP	12	22	186,6
5	SP	12	17	236,3
6	SP	12	9	70,8

Fonte: BDMEP

2.1.1.1 Valores *missings*

Consiste na ausência de valores, que pode ter sido provocada por falhas humanas, de software ou de hardware, ou simplesmente o valor não estava disponível no momento da captura (SILVA, 2011). A Tabela 2 contém um valor *missing* no registro com id 0 referente ao atributo NumDiasPrecipitação.

2.1.1.1.1 Registros missings

Uma outra forma de *missing* é quando se verifica a possível ausência de registros ao analisar o comportamento de um atributo, por exemplo, na Tabela 2, se analisarmos o atributo id, ele tem comportamento sequencial e discreto, podemos notar que é possível que haja um registro faltante referente ao id 2.

2.1.1.2 Valores inválidos

Valores inválidos na maioria das vezes estão relacionados com atributos quantitativos, pois estes precisam seguir uma notação numérica,

caso contrário o valor que deveria corresponder a um número se torna um valor inválido. Este tipo de problema geralmente é causado, por exemplo, devido a falhas de transcrição, diferentes notações numéricas dependendo da região. Exemplo, “12,345.67”é uma notação numérica válida no Canadá, porém não no Brasil. Na Tabela 2, o atributo Mês pode ser representado numericamente ou textualmente, dessa forma seu tipo pode ser qualitativo ou quantitativo discreto, se for considerado quantitativo discreto o valor “Dezembro”é inválido, já que “Dezembro”não é um valor numérico.

Atributos qualitativos também possuem valores inválidos, porém as categorias de um atributo qualitativos podem expressar qualquer coisa, desta forma automaticamente não podem ser detectados.

2.1.1.3 Valores *outliers*

Outlier, valor aberrante, valor atípico ou ponto discrepante é um valor que apresenta um grande afastamento dos demais (SILVA, 2011).

Por exemplo, a sociedade brasileira é separada em várias classes sociais, de modo que os indivíduos de uma mesma classe tenham condições econômicas semelhantes, supondo que Bill Gates morasse no Brasil, com certeza ele seria um exemplo de *outlier* caso erroneamente fosse classificado pertencer a classe média, uma vez que sua condição financeira está bem acima do padrão esperado dentro da classe. Mais à frente serão apresentadas dois modos de detectar *outliers* para atributos quantitativos. Na Tabela 2, o valor 836,6 de PrecipitaçãoTotal é um provável *outlier*.

Tabela 2: Exemplo de *missings*, valores inválidos e *outliers*

id	Estação	Mês	NumDiasPrecipitação	PrecipitaçãoTotal
0	RS	12		134,1
1	RS	12	5	48,2
3	SC	12	13	835,6
4	SP	12	22	186,6
5	SP	Dezembro	17	236,3
6	SP	12	9	70,8

2.1.2 Medidas descritivas

Existem medidas que servem para descrever resumidamente os atributos quantitativos (BARBETTA; REIS; BORNIA, 2010). As mais utilizadas serão apresentadas nos tópicos a seguir. Considere $\{x_1, x_2, \ldots, x_N\}$ uma população com N indivíduos.

2.1.2.1 Valor máximo e mínimo

São os valores extremos (máximo e mínimo) de um atributo, Equações 2.2 e 2.1 respectivamente.

$$max = máximo\{x_1, x_2, \ldots, x_N\} \tag{2.1}$$

$$min = mínimo\{x_1, x_2, \ldots, x_N\} \tag{2.2}$$

2.1.2.2 Média aritmética simples

Valor típico ou ponto de equilíbrio de um atributo.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{2.3}$$

2.1.2.3 Amplitude total

Forma simples de representar a dispersão, pode levar a conclusões errôneas quando existem valores discrepantes.

$$r = max - min \tag{2.4}$$

2.1.2.4 Desvio padrão

Comumente utilizado para representar dispersão em relação à média. A Equação 2.5 considera todos os indivíduos de uma população, caso se tenha uma amostra de tamanho n da população deve-se usar a Equação 2.6 considerando $\{x_1, x_2, \ldots, x_n\}$ como sendo os indivíduos

da amostra e $\overline{x}$ a média da amostra.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \tag{2.5}$$

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2} \tag{2.6}$$

Neste trabalho foi considerado apenas o desvio padrão da população.

2.1.2.5 Coeficiente de variação

Expressa a variabilidade dos valores de um atributo sem considerar a ordem de grandeza do atributo.

$$c_v = \frac{\sigma}{\mu} \tag{2.7}$$

2.1.2.6 Quartis

Divide a população em quatro partes iguais, o cálculo de cada quartil deve ser feito utilizando a população ordenada ascendentemente. Dependendo do valor de N, o cálculo de cada quartil é realizado utilizando interpolação linear. A operação $x \; mod \; y$ significa o resto da divisão inteira de x por y.

- Primeiro quartil ou quartil inferior (q_i), é o valor que separa os 25% menores valores dos 75% maiores valores da população.

$$\text{Se } (N+3) \bmod 4 = 0, \text{ então } q_i = x_{\frac{N+3}{4}}$$

$$\text{Senão, } q_i = x_{\left\lfloor \frac{N+3}{4} \right\rfloor} + \frac{(N+3) \bmod 4}{4} * \left(x_{\left\lceil \frac{N+3}{4} \right\rceil} - x_{\left\lfloor \frac{N+3}{4} \right\rfloor} \right) \tag{2.8}$$

- Mediana, valor que ocupa a posição central da população.

$$\begin{aligned} &\text{Se } N \text{ é ímpar } m_d = x_{\frac{N+1}{2}} \\ &\text{Senão, } m_d = \frac{x_{\frac{N}{2}} + x_{\frac{N+2}{2}}}{2} \end{aligned} \tag{2.9}$$

- Terceiro quartil ou quartil superior (q_s), valor que separa os 75% menores valores dos 25% maiores valores da população.

$$\begin{aligned} &\text{Se } (3N+1) \bmod 4 = 0, \text{ então } q_s = x_{\frac{3N+1}{4}} \\ &\text{Senão, } q_s = x_{\left\lfloor \frac{3N+1}{4} \right\rfloor} + \frac{(3N+1) \bmod 4}{4} * \left(x_{\left\lceil \frac{3N+1}{4} \right\rceil} - x_{\left\lfloor \frac{3N+1}{4} \right\rfloor} \right) \end{aligned} \tag{2.10}$$

2.1.2.7 Desvio interquartílico

É calculado através da diferença entre o quartil inferior e superior. Quanto mais dispersa for a distribuição dos valores de um atributo, maior será seu desvio interquartílico.

$$d_q = q_s - q_i \tag{2.11}$$

2.1.3 Distribuição de frequências

As contagens de indivíduos que se enquadram a um valor ou intervalo (classe) formam a distribuição de frequências do atributo (BARBETTA; REIS; BORNIA, 2010). Algumas características podem ser identificadas a partir da distribuição de frequências: valores ou intervalos que ocorrem com mais/menos frequência e possíveis *outliers*.

A maneira como é construída difere para atributos qualitativos e quantitativos, ambas serão apresentadas nas duas Seções a seguir.

2.1.3.1 Distribuição de frequências para atributos qualitativos

Consiste em contar as ocorrências de cada categoria do atributo qualitativo (BARBETTA; REIS; BORNIA, 2010). A Tabela 3 é um exem-

plo da distribuição de frequências do atributo Estação da Tabela 1.

Tabela 3: Frequências do atributo Estação

Estação	Frequência
SC	2
RS	2
SP	3

2.1.3.2 Distribuição de frequências para atributos quantitativos

A forma de construir a distribuição de frequências para atributos quantitativos se divide entre os atributos quantitativos discretos e contínuos, como apresentadas a seguir.

2.1.3.2.1 Atributos discretos

A distribuição de frequências dos atributos discretos pode ser construída de forma análoga à atributos qualitativos, porém não são contadas as ocorrências das categorias, mas sim dos valores (BARBETTA; REIS; BORNIA, 2010), exemplo Tabela 4.

Tabela 4: Frequências do atributo Mês

Mês	Frequência
12	7

2.1.3.2.2 Atributos contínuos

A construção da distribuição de frequências dos atributos quantitativos contínuos é realizada através da divisão do intervalo $[min, max]$ do atributo desejado em k subintervalos (classes) de amplitude h. A quantidade de classes k para compor a distribuição de frequências é uma escolha arbitrária (BARBETTA; REIS; BORNIA, 2010). Existem técnicas que tentam estimar o número de classes. Ainda considerando N a quantidade de indivíduos da população, algumas técnicas são:

- Dada uma quantidade de classes k qualquer, a amplitude h das classes é calculada pela Equação 2.12.

$$h = \frac{max - min}{k} \tag{2.12}$$

- Dada uma amplitude da classe h arbitrária, a quantidade de classe k é calculada pela Equação 2.13.

$$k = \left\lceil \frac{max - min}{h} \right\rceil \tag{2.13}$$

- Amplitude de Scott (2009), onde σ é o desvio padrão da população.

$$h = \frac{3,5 * \sigma}{\sqrt[3]{N}} \tag{2.14}$$

- Quantidade de classes de Hines, Montgomery e Borror (2008).

$$k = \sqrt{N} \tag{2.15}$$

- Quantidade de classes de Sturges (1926).

$$k = \lceil \log_2 N \rceil + 1 \tag{2.16}$$

Neste trabalho foram utilizados os estimadores de quantidade de classes de Scott e Hines, os mesmo utilizados no E&P Risk, de modo que para $N > 200$ usa-se Scott, caso contrário Hines, já que segundo Scott (2009), o estimador Sturges não resulta em boas classes para gerar histogramas. Mesmo utilizados estes dois estimadores ainda foi definido uma regra na qual 25 é o número máximo de classes, mais classes que isso acaba tornando difícil de se obter informações do histograma.

Definido a quantidade de classes (k) e a amplitude das classes (h), as classes são criadas de forma que englobem todos o valores do atributo.

Construindo a tabela de frequências do atributo PrecipitaçãoTotal da Tabela 1 por Sturges:

$$min = 36,6 \tag{2.17}$$

$$max = 236,3 \tag{2.18}$$

$$k = \lceil \log_2 n \rceil + 1 = \lceil \log_2 7 \rceil + 1 = 4 \tag{2.19}$$

$$h = \frac{max - min}{k} = \frac{236,3 - 36,6}{4} = 50,175 \tag{2.20}$$

$$C = \{35,6 \vdash 85,775; 85,775 \vdash 135,95; 135,95 \vdash 186,125; 186,125 \vdash 236,3\} \tag{2.21}$$

Contando a frequência de cada classe (Tabela 5).

Tabela 5: Frequências das classes do atributo PrecipitaçãoTotal

PrecipitaçãoTotal	Frequência
$35,6 \vdash 85,775$	3
$85,775 \vdash 135,95$	1
$135,95 \vdash 186,125$	1
$186,125 \vdash 236,3$	2

2.1.4 Representações gráficas

Nesta Seção serão apresentas quatro formas de representação dos dados e suas principais características.

2.1.4.1 Histograma e gráfico de barras

Além das tabelas de frequências da Seção 2.1.3, o histograma e o gráfico de barras (ou colunas), são alternativas de se apresentar graficamente a distribuição de frequências de um atributo (BARBETTA; REIS; BORNIA, 2010). Em geral de forma mais sugestiva.

Em ambos o comprimento, geralmente medido pelo eixo y, corresponde as frequências absolutas (ou relativas) dos valores ou intervalos dependendo do tipo do atributo, esses valores ou intervalos são geralmente representados pelo eixo x (CHAMBERS, 1983). Apesar do histograma também ser um gráfico de barras, ele se difere no fato de que suas barras estão justapostas, já que ele representa o intervalo que engloba todos os valores de um atributo quantitativo contínuo.

As Figuras 1, 2 são exemplos de gráfico de barras de atributos qualitativos e quantitativos discretos respectivamente. Enquanto a Figura 3 é exemplo de histograma de atributos quantitativos contínuos.

Figura 1: Gráfico de barras (qualitativo)

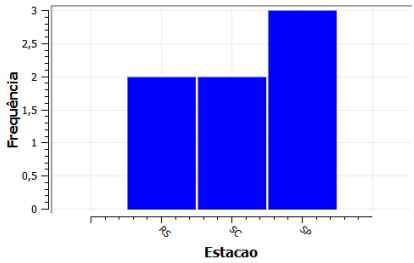


Figura 2: Gráfico de barras (quantitativo discreto)

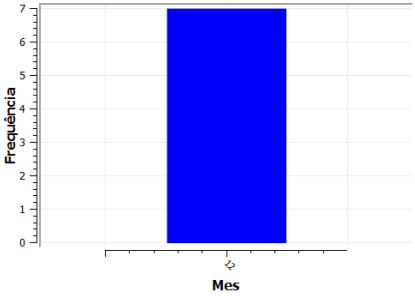
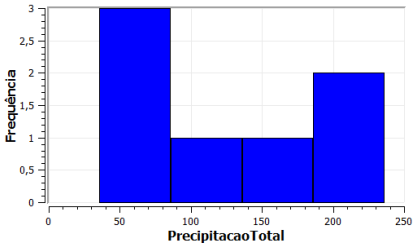


Figura 3: Histograma (quantitativo contínuo)



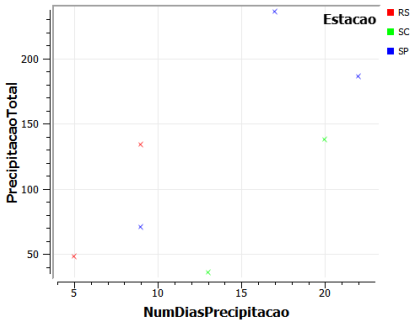
2.1.4.2 Diagrama de dispersão

O diagrama de dispersão é um dos mais utilizados para análise dos dados, possibilita identificar as correlações entre os atributos (CHAMBERS, 1983). Pode representar os valores de $n \geq 2$ atributos.

Os valores dos atributos são organizados em tuplas com n componentes que correspondem a cada um dos eixos do gráfico respectivamente, deste modo, cada tupla representa uma coordenada ou ponto no gráfico.

Com objetivo de diminuir o total de dimensões do gráfico com mais de duas dimensões, pode-se substituir algumas delas de modo que os valores da dimensão substituída sejam representados na forma, por exemplo, de cor ou formato do ponto. A Figura 4 é um exemplo de gráfico de dispersão.

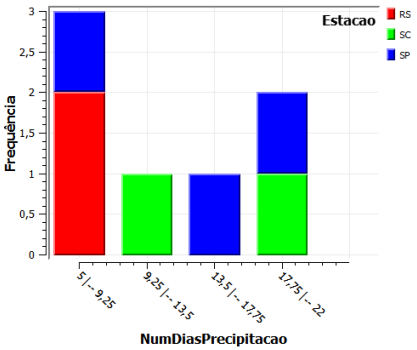
Figura 4: Diagrama de dispersão



2.1.4.3 Gráfico de barras empilhadas ou agrupadas

É uma extensão do gráfico de barras apresentado na Seção 2.1.4.1, porém permite representar frequências conjuntas de um grupo de atributos, geralmente dois. A apresentação das frequências conjuntas destes atributos é realizada de forma que cada coluna, visualmente identificada, represente a frequência de uma combinação de valores dos atributos. A Figura 5 é um exemplo de gráfico de barras empilhadas.

Figura 5: Gráfico de barras empilhadas

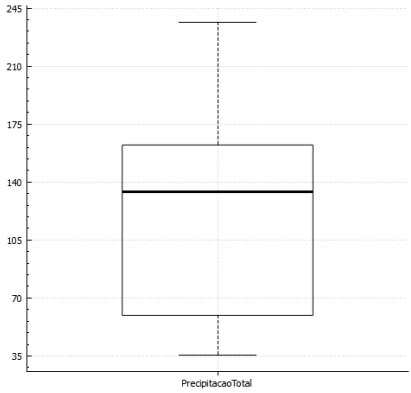


2.1.4.4 Diagrama de caixa

O diagrama de caixa ou *boxplot* permitem apresentar um resumo da distribuição dos valores de um atributo (BARBETTA; REIS; BORNIA, 2010).

É composto por uma caixa que corresponde ao desvio interquartílico de modo a englobar 50% dos valores da distribuição. A Figura 6 apresenta o diagrama de caixa do atributo PrecipitaçãoTotal da Tabela 1. As arestas horizontais na parte superior, central e inferior da caixa correspondem aos quartis superior, mediana e inferior, respectivamente.

Figura 6: Diagrama de caixa



Os seguimentos de retas na vertical, denominados *whisker*, que estão conectadas ao topo e base da caixa se estendem até os valores calculados pelas Equações 2.22 e 2.23 respectivamente.

$$w_s = m\acute{i}nimo\{q_s + 1,5 * d_q; max\} \tag{2.22}$$

$$w_i = máximo\{q_i - 1,5 * d_q; min\} \tag{2.23}$$

O diagrama de caixa é uma maneira fácil de identificar *outliers*. Os valores que são maiores que w_s ou menores que w_i são considerados *outliers* e são representados geralmente no gráfico de bolinhas.

2.1.5 Identificação de valores *outliers*, inválidos ou *missings*

Está Seção descreve como identificar os principais fatores que influenciam a má qualidade dos dados, que são os valores *outliers*, inválidos e *missings*.

2.1.5.1 Identificação de *outliers*

Como já definido, *outliers* são valores fora do padrão. Existem diversas técnicas para identificação destes, porém neste trabalho serão apresentadas nas Seções a seguir as duas técnicas comumente utilizadas.

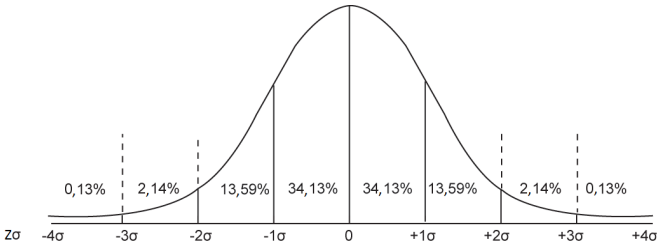
2.1.5.1.1 Teste Z-score

Z-score é uma medida de dispersão, que pode assumir valores negativos ou positivos descrevendo o deslocamento de um determinado valor em relação à média através de z desvios padrões, Equação 2.24, onde x é o valor que se deseja calcular o seu respectivo *z-score*.

$$z = \frac{x - \mu}{\sigma} \tag{2.24}$$

O teste de *z-score* consiste em considerar valores que tem seus respectivos *z-scores* fora de um intervalo $[z_{min}, z_{max}]$ (SILVA, 2011). Este teste funciona bem quando a distribuição dos valores de um atributo se aproximam de uma distribuição normal, por exemplo, de acordo com a Figura 7, um intervalo [-2 , 2] engloba 95,44% do valores existentes, os demais 4,56% poderiam ser considerados *outliers*.

Figura 7: *Z-score*



2.1.5.1.2 Diagrama de caixa

Como foi apresentado, o diagrama de caixa além de apresentar a dispersão dos valores de um atributo também permite identificar possíveis valores *outliers*, de modo que aqueles que estejam fora do intervalo $[w_i, w_s]$ são considerados *outliers* (BARBETTA; REIS; BORNIA, 2010).

2.1.5.2 Identificação de valores inválidos

Supondo que os dados sejam provenientes de algo que os armazene em forma de texto e deseja-se realizar uma análise destes através de um computador. Computacionalmente, textos são textos independentemente se eles representam números o computador não consegue compreender, deste modo valores de atributos considerados quantitativos em forma de texto precisam ser convertidos para tipo numérico de modo que o computador consiga compreender. Assim um valor é considerado inválido se a sua representação textual não pode ser convertida em um valor numérico entendível por computadores.

Já para os valores inválidos dos atributos qualitativos, não podem ser identificados automaticamente desta mesma forma, já que as categorias destes tipos de atributos podem expressar qualquer coisa, uma solução seria apresentar as categorias presentes no atributos e alguém ou algo definir o que é válido ou não.

2.1.5.3 Identificação de valores e registros *missings*

Conforme apresentado, *missing* é a ausência de valor, a identificação deste é feita simplesmente por uma verificação, se o valor não contém nada, então ele é um valor *missing*.

Já a identificação dos registros *missings* pode ser realizada observando o comportamento de um atributo X, sabendo que todos os valores de X são discretos, podemos considerar que existe um registro *missing* quando existe um valor discreto pertencente ao intervalo $[X_{min}, X_{max}]$ porém esse valor não pertence ao conjunto de valores de X, resumindo, consiste em identificar valores que possivelmente deveriam estar dentro do intervalo de valores do atributo X, para todos esses valores identificados, é provável que exista um registro *missing*.

2.2 TRATAMENTO DOS DADOS

O tratamento de valores que são *outliers*, inválidos ou *missings* ou qualquer outro valor considerado inadequado pode ser realizado de maneira semelhante, podendo:

- Remover o registro, entretanto pode resultar na perda de informações dos outros atributos.
- Substituir o valor inadequado por um outro valor, como por exemplo a média aritmética da população ou parte dela.
- Identificar uma distribuição de probabilidades que melhor corresponda ao comportamento dos valores do atributo, e a partir desta distribuição gerar um valor aleatório que substitua o valor inadequado.
- Para atributos correlacionadas, pode-se fazer uso, por exemplo, de técnicas de aprendizagem de máquina, como regressões, que permitem representar as correlações entre atributos, de modo a prever qual o valor mais adequado, que condiga com o comportamento da relação, deve substituir o valor incorreto.

2.3 DISTRIBUIÇÕES DE PROBABILIDADES

Uma DP (Distribuição de probabilidades) é uma função matemática que descreve a probabilidade de cada valor de um conjunto

possível de valores ocorrer (FREITAS FILHO, 2008) Uma DP é utilizada para representar o comportamento aleatório de um atributo. Inúmeras são as distribuições, porém as utilizadas neste trabalho foram: Normal (μ, σ), Lognormal (μ, σ), Uniforme (a, b), Triangular (a, b, m), Exponencial (β), Gamma (α, β), Weibull (α, β) e Beta (α, β).

Segundo Freitas Filho (2008), a identificação da DP segue como:

- A construção da distribuição de frequências e a utilização de histogramas são muito úteis para um inferência inicial de qual DP utilizar.
- Escolhida a DP é necessário estimar seus parâmetros. A maioria das distribuições utilizam como parâmetros as medidas descritivas ou parâmetros calculados a partir delas.

  Os parâmetros estimados para $\mu, \sigma, a, b, m, \alpha$ e β são respectivamente as Equações 2.3, 2.5, 2.2, 2.1, 2.25, 2.27 e 2.26. Equações 2.27 e 2.26 por Law, Kelton e Kelton (1991)

$$m = 3\mu - (min + max) \tag{2.25}$$

$$\beta = \frac{\sigma^2}{\mu} \tag{2.26}$$

$$\alpha = \left(\frac{\mu}{\sigma}\right)^2 \tag{2.27}$$

  Uma outra maneira de estimar os parâmetros da distribuição é utilizando estimador de máxima verossimilhança (MLE), na qual foi utilizada na implementação da biblioteca de funções lib_fit desenvolvida por Formighieri (2007), mais detalhes sobre serão apresentados na Seção 3.3.1.

2.3.1 Testes de aderência

Os testes de aderência têm como objetivo verificar a qualidade da DP. De forma a observar o desvio entre a distribuição amostral e a teórica.

2.3.1.1 Teste chi-quadrado

Dado dois conjuntos de valores, um composto pelas amostras observadas $O = \{o_1, o_1, \ldots, o_N\}$ e o outro pelas amostras esperadas

$E = \{e_1, e_1, \ldots, e_N\}$. Para realizar o teste chi-quadrado é necessário construir as distribuições de frequências fo e fe para os conjuntos O e E respectivamente, Seção 2.1.3, com pelo menos 5 classes (ou intervalos). O cálculo do valor chi-quadrado é realizado utilizando a Equação 2.28, onde k é o total de classes.

$$\mathcal{X}^2 = \sum_{i=1}^{k} \frac{(fo_i - fe_i)^2}{fe_i} \tag{2.28}$$

Se as frequências observadas e esperadas estiverem próximas, o numerador da equação será pequeno e consequentemente $\mathcal{X}^2$ também. Quanto menor o valor de $\mathcal{X}^2$, maior a probabilidade da DP escolhida condizer com o comportamento real.

Outra forma de comparação pode ser feita utilizando o valor-p, que permite testar se hipótese de que os valores observados e esperados se coincidem é nula. Quanto maior for o valor de p, maior a probabilidade de a hipótese ser falsa.

2.4 REGRESSÕES

As regressões são métodos que permitem identificar a relação entre o valor de um atributo y (dependente) e o valor de outros atributos $x, z, \ldots, t$ (independentes) de modo que permita prever ou calcular o valor de y em função dos valores de $x, z, \ldots, t$.

As relações entre os atributos podem ser de vários tipos: linear, exponencial, logarítmica, potência, logística, etc. Neste trabalho somente será abordada a regressão linear simples.

2.4.1 Regressão linear simples

A regressão linear simples (RLS) é a regressão mais básica de todas, descreve a relação linear, em forma de reta, entre dois atributos x e y, independente e dependente respectivamente (REIS, 2008). A Equação 2.29 determina a relação entre ambos os atributos, ou seja, o cálculo de y em função de x. O e é um valor residual aleatório que inclui outros fatores, além de x, que podem influenciar o valor de y, α representa a interceptação da reta com o eixo vertical e β é coeficiente angular da reta.

$$y = \alpha + \beta * x + e \tag{2.29}$$

Cálculo das constantes β e α, Equações 2.30 e 2.31 respectivamente, considerando $\{x_1, x_2, \ldots, x_N\}$ e $\{y_1, y_2, \ldots, y_N\}$ os valores dos atributos x e y respectivamente que se deseja identificar a relação.

$$\beta = \frac{N * \sum_{i=1}^{N} (x_i * y_i) - \sum_{i=1}^{N} x_i * \sum_{i=1}^{N} y_i}{N * \sum_{i=1}^{N} x_i^2 - \left(\sum_{i=1}^{N} x_i \right)^2} \tag{2.30}$$

$$\alpha = \mu_y - \beta * \mu_x \tag{2.31}$$

3 DESENVOLVIMENTO

Neste capítulo serão apresentados detalhes sobre a codificação da ferramenta.

3.1 METODOLOGIA DE TRABALHO

A criação desta ferramenta consistiu basicamente em codificar os conceitos apresentados no Capítulo 2 e desenvolver a GUI (Graphical user interface), que permita o usuário utilizar as funcionalidades implementadas.

Para criar a ferramenta foi alocada uma equipe formada apenas pelo autor e orientador deste trabalho.

As atividades consistiam em estudar os conceitos necessários, solucionando eventuais dúvidas junto ao orientador e então traduzi-los para uma linguagem de programação. A atividade final e a mais trabalhosa foi o desenvolvimento da interface.

A linguagem de programação escolhida foi C++, na qual o autor deste trabalho tem maior experiência, além de apresentar um bom desempenho nos mais diversos problemas. Foram utilizadas as seguintes ferramentas de desenvolvimento:

- Microsoft Visual Studio 2010 Ultimate x86 (MVS): Ambiente de desenvolvimento (IDE).
- Qt 5.0.2 x86 MVS 2010 OpenGL: Ambiente de desenvolvimento (IDE).
- Qt Visual Studio Add-in 1.2.2: Integra ao MVS a maioria das funcionalidades disponibilizadas pelo Qt.

3.2 ORGANIZAÇÃO E FUNCIONAMENTO

Nesta Seção serão apresentados os principais aspectos de organização e funcionamento da ferramenta.

A ferramenta foi dividida em 6 subprojetos:

- lib_utilidades: Biblioteca que contém funções variadas que servem de apoio para os outros projetos.

- lib_qwt e lib_qcustomplot: Bibliotecas externas para plotar gráficos.
- lib_fit: Biblioteca que tem como objetivo estimar parâmetros de uma DP dentre as que são suportadas a partir de um conjunto de valores de um atributo, calcular testes de aderência, assim como utilizar a DP.
- lib_analise_tratamento: Biblioteca criada neste trabalho contendo funções de coleta de estatísticas, integração das DP fornecidas pela lib_fit, geração das informações necessárias para os gráficos, regressões lineares e gerenciamento dos registros e seus atributos.
- gui_analise_tratamento: É o ponto de inicialização da ferramenta e contém o código da interface.

Todo o código fonte dos subprojetos, exceto lib_qwt e lib_qcustomplot que são facilmente encontradas na internet, pode ser encontrado no Anexo B.

Nas Seções a seguir serão apresentados mais detalhes dos dois principais subprojetos trabalhados referentes a parte interna da aplicação, são eles: lib_fit e lib_analise_tratamento.

3.3 ASPECTOS PRINCIPAIS DE IMPLEMENTAÇÃO

Nesta Seção serão apresentados os principais aspectos de implementação relacionados aos subprojetos lib_fit e lib_analise_tratamento.

3.3.1 lib_fit

Como já dito, a lib_fit é uma biblioteca que fornece um conjunto de funções relacionadas com DPs. As seguintes funcionalidades disponibilizadas serão utilizadas pela ferramenta desenvolvida:

- A partir de um conjunto de valores de um atributo estimar os parâmetros de uma DP utilizando estimador de máxima verossimilhança (MLE).
- Gerar uma DP inserindo seus parâmetros manualmente.
- Realizar testes de aderência através do valor-p.
- Gerar um valor aleatório de acordo com uma DP criada.

A lib_fit suporta os seguintes tipos de DPs: Normal, Lognormal, Uniforme, Triangular, Exponencial, Gamma, Weibull e Beta.

3.3.2 lib_analise_tratamento

Nesta Seção serão apresentadas as principais classes e suas características da biblioteca desenvolvida neste trabalho para ATD. Considere as seguintes detalhes:

- Somente os aspectos mais importantes foram apresentados, assim nem todo o código das classes foram apresentados.
- A maioria das funções de construção, destruição, obtenção (*gets*), atribuição (*sets*), entre outras e seus parâmetros foram omitidos com "...".
- Muitas das funções parecem não ter retorno, pois dizem retornar *void*, porém por questões de desempenho a maioria dos resultados destas funções são retornados através de referências por parâmetro.

3.3.2.1 Classe *ValorBase*

Classe abstrata, Listagem 3.1, que contém os atributos de classe *_missing* e *_valorStr*, que respectivamente significam se o valor é *missing* e sua representação em forma de texto.

As funções virtuais puras, nas quais as classes herdadas são obrigadas a implementarem, são utilizadas para comparação entre dois valores e para clonar o próprio. Funções de obtenção ou construção foram omitidas por "...".

```
1 class ValorBase
2 {
3 public:
4     ...
5     virtual bool operator <(const ValorBase &outro) const =
          0;
6     virtual bool operator ==(const ValorBase &outro) const =
          0;
7     virtual ValorBase *clone() const = 0;
8 protected:
9     bool _missing;
```

```
10      string _valorStr;
11  };
```

Listagem 3.1: Classe *ValorBase*

3.3.2.1.1 Classe ValorQualitativo

É uma classe estendida de *ValorBase*. Nenhum atributo de classe foi acrescentado ou removido, apenas foram implementadas as funções para comparação entre objetos a e b que são das classes *ValorQualitativo* e *ValorBase* respectivamente.

3.3.2.1.2 Classe Valor Quantitativo

Também é uma classe estendida de *ValorBase*. Foram acrescentados os atributos de classe $\_valorDbl$ e $\_formatoValido$ que representam respectivamente o valor como um número (computacionalmente) e se o formato do mesmo é válido. Assim como a classe *ValorQualitativo*, foram implementados os métodos para comparação entre objetos a e b que são das classes *ValorQuantitativo* e *ValorBase* respectivamente, tais funções, funções de obtenção e de construção foram omitidas na Listagem 3.2 por "...".

```
1  class ValorQuantitativo : public ValorBase
2  {
3  public:
4      ...
5  private:
6      double _valorDbl;
7      bool _formatoValido;
8  };
```

Listagem 3.2: Classe *ValorQuantitativo*

3.3.2.2 Classe *Registro*

Classe *Registro* representa uma linha da tabela de dados, e contém um vetor de valores que podem ser qualitativos ou quantitativos. Funções de obtenção, atribuição, construção e destruição foram omitidas por

"...".

```
class Registro
{
public:
    ...
    bool algumValorInvalidoOuMissingNosIndices(...) const;
    ...
private:
    vector<ValorBase *const> _valores;
};
```

Listagem 3.3: Classe *Registro*

3.3.2.3 Classe *RegressãoLinearSimples*

Classe que representa uma RLS, apresentada na Seção 2.4.1, Listagem 3.4, funções de obtenção, construção padrão e destruição foram omitidas por "...".

A construção da RLS é realizada pelo próprio construtor da classe, onde β e α são calculados com as Equações 2.30 e 2.31 respectivamente.

A Equação 2.29 da RLS é implementada pela função *gerarValorDependente*, na qual gera um valor dependente em função de um independente.

```
class RegressaoLinearSimples
{
public:
    ...
    RegressaoLinearSimples(const vector<double> &
        valoresIndependentes, const vector<double> &
        valoresDependentes);
    double gerarValorDependente(const double
        valorIndependente) const;
    ...
private:
    double _alpha;
    double _beta;
};
```

Listagem 3.4: Classe *RegressãoLinearSimples*

3.3.2.4 Classe *Distribuição*

Classe com objetivo de integrar a lib_fit, Listagem 3.5. Funções de obtenção, construção, destruição e comparação foram omitidas com "...".

A classe permite criar qualquer distribuição suportada pela lib_fit a partir de parâmetros ou de um conjunto de valores. Também se pode criar distribuições somente a partir do tipo dela, neste caso os parâmetros serão os padrões.

```
1  class Distribuicao
2  {
3  public:
4      ...
5      double gerarValorAleatorio() const;
6      ...
7
8  private:
9      fit::base_distrib *_fitDistribuicao;
10     fit::_distribution_type _tipo;
11     double _valorP;
12     bool _valorPValido;
13
14     void gerarDistribuicaoDeValores(...);
15     void gerarDistribuicaoDeParametros(...);
16     void calcularValorP(...);
17 };
```

Listagem 3.5: Classe *Distribuição*

3.3.2.4.1 Classe GeradorDistribuições

Está classe, Listagem 3.6, serve de apoio a classe *Distribuição*, sua função é gerar, a partir de um conjunto de valores de um atributo, todas as distribuições suportadas pela lib_fit, de modo que as DPs sejam separadas em duas listas, geradas com sucesso e as com falhas. A lista de DPs construídas com sucesso é ordenada descendente de acordo com o valor-p, de modo que a primeira DP é a que melhor aderiu aos valores de um atributo.

```
1  class GeradorDistribuicoes
2  {
3  public:
```

```
4     static void gerarTodasDistribuicoes(...);
5 private:
6     static const fit::_distribution_type tiposDistri[8];
7 };
```

Listagem 3.6: Classe *GeradorDistribuições*

3.3.2.5 Classe *TabelaDeRegistros*

A classe *TabelaDeRegistros*, Listagem 3.7, contém os registros, atributos e tipos dos atributos (qualitativo ou quantitativo contínuo ou discreto). Cada registro é identificado por um id e são armazenados em *_registros* na Linha 9. Os atributos e seus tipos são armazenados em *_atributos* e *_tipos*, Linhas 10 e 11, respectivamente.

As principais funções da classe são listadas abaixo, estas funções e outras de construção e destruição foram omitidas por "...":

- Gerenciar registros, atributos e seus tipos.
- Construir instâncias das classes estendidas de *ValorBase* de acordo com os tipos dos atributos.
- Preencher coluna ou registro utilizando RLS ou DP.
- Identificar ou remover registros por atributo que tenham valor:
  - *Missing* ou inválido.
  - Dentro de um intervalo.
  - Igual a outro valor.
- Substituir valores que podem ser identificados pelo item acima por um valor qualquer, utilizando DP ou RLS.
- Verificar se um atributo pode ser utilizado como referência para identificar possíveis registros *missings*, Seção 2.1.1.1.1.

```
1 class TabelaDeRegistros
2 {
3 public:
4     typedef map<size_t, Registro> MapRegistros;
5     typedef tuple<double, double, Operacao> Intervalo;
6     ...
7
8 private:
```

```
9       MapRegistros _registros;
10      vector<string> _atributos;
11      vector<TipoDado> _tipos;
12      size_t _totalRegistro;
13      size_t _novoRegistroIndice;
14      ...
15  };
```

Listagem 3.7: Classe *TabelaDeRegistros*

3.3.2.6 Classe *ColetorEstatísticas*

A classe *ColetorEstatísticas* é responsável por coletar estatísticas dos atributos, são elas:

- Medidas descritivas.
- Distribuições de frequências.

Mais detalhes serão apresentados nas Seções a seguir.

3.3.2.6.1 Tipos definidos

Para facilitar o entendimento do que é cada atributo da classe *ColetorEstatísticas*, algumas estruturas foram renomeadas, Listagem 3.8, são elas:

- *ClasseQuant*, Linha 4: Classe ou intervalo de um atributo quantitativo contínuo.
- *FrequenciasQuantContinuo*, Linha 5: Frequências de cada classe ou intervalo de um atributo quantitativo contínuo.
- *FrequenciasQuantDiscreto*, Linha 6: Frequências de cada valor de um atributo quantitativo discreto, porém também funciona para os contínuos.
- *FrequenciasQuali*, Linha 7: Frequências de cada categoria de um atributo qualitativo.

```
1  class ColetorEstatisticas
2  {
3  public:
```

```
4       typedef pair<double, double> ClasseQuant;
5       typedef map<ClasseQuant, int> FrequenciasQuantContinuo;
6       typedef map<double, int> FrequenciasQuantDiscreto;
7       typedef map<string, int> FrequenciasQuali;
8   };
```

Listagem 3.8: Tipos definidos da classe *ColetorEstatísticas*

3.3.2.6.2 Constantes

Foram definidas algumas constantes, Listagem 3.9, são elas:

- $\_FATOR\_WHISKER\_SUPERIOR$, Linha 4: É o fator $1,5$ da Equação 2.22.
- $\_FATOR\_WHISKER\_INFEIROR$, Linha 5: É o fator $1,5$ da Equação 2.23.
- $\_ZMIN$, Linha 6: Valores com *z-score* menor que $z_{min} = -2,5$ são considerados *outliers*.
- $\_ZMAX$, Linha 7: Valores com *z-score* maior que $z_{max} = 2,5$ são considerados *outliers*.

Todas as constantes apresentadas estão relacionadas com parâmetros para identificação de *outliers*, porém a classe *ColetorEstatísticas* contém funções para identificar *outliers* que permitem outros parâmetros.

```
1   class ColetorEstatisticas
2   {
3   public:
4       static const double _FATOR_WHISKER_SUPERIOR;
5       static const double _FATOR_WhISKER_INFEIROR;
6       static const double _ZMIN;
7       static const double _ZMAX;
8   };
```

Listagem 3.9: Constantes da classe *ColetorEstatísticas*

3.3.2.6.3 Estruturas

Foram criadas duas estruturas para representarem as estatísticas dos atributos, Listagem 3.10, são elas:

- *EstatisticaGeral*, Linha 4: são medidas descritivas que se aplicam a ambos atributos qualitativos e quantitativos.
  - *missings*: Total de valores *missings* presentes no atributo.
  - *distintos*: Tamanho do conjunto de valores formado pelos valores sem repetição de um atributo.
  - *únicos*: Total de valores que são únicos dentre os valores do atributo, ou seja, valores que aparecem somente uma vez.
- *EstatisticaQuant*, Linha 11: são medidas descritivas, Seção 2.1.2, e outras informações auxiliares, listadas a seguir, que se aplicam apenas aos atributos quantitativos:
  - Dois vetores com os valores *outliers* identificados utilizando o teste *z-score* e o diagrama de caixa, com os parâmetros constantes mostrados na Seção 3.3.2.6.2.
  - Total de valores inválidos ou *missings*.
  - Total de valores válidos.

```
1  class ColetorEstatisticas
2  {
3  public:
4      struct EstatisticaGeral
5      {
6          int missings;
7          int distintos;
8          int unicos;
9      };
10
11     struct EstatisticaQuant
12     {
13         double media;
14         double maximo;
15         double whiskerSuperiorDiagramaCaixa;
16         double quartilSuperior;
17         double mediana;
18         double quartilInferior;
19         double whiskerInferiorDiagramaCaixa;
20         double minimo;
21         double desvioPadrao;
22         double coeficienteVariacao;
23         double amplitudeTotal;
24         double desvioInterquartilico;
25
26         vector<double> outliersDiagramaCaixa;
27         vector<double> outliersZScore;
28
```

```
29          int totalValoresInvalidosOuMissings;
30          int totalValoresValidos;
31      };
32  };
```

Listagem 3.10: Estruturas da classe *ColetorEstatísticas*

3.3.2.6.4 Atributos

Os atributos da classe *ColetorEstatísticas* são as frequências e estatísticas de cada atributo de acordo com o tipo do mesmo.

```
1  class ColetorEstatisticas
2  {
3  private:
4      map<string, EstatisticaGeral> _estatisticasGerais;
5      map<string, EstatisticaQuant> _estatisticasQualitativos;
6      map<string, FrequenciasQuali> _frequenciasQuali;
7      map<string, FrequenciasQuantDiscreto>
           _frequenciasQuantDisc;
8      map<string, FrequenciasQuantContinuo>
           _frequenciasQuantCont;
9  };
```

Listagem 3.11: Atributos da classe *ColetorEstatísticas*

3.3.2.6.5 Funções

As principais funções da classe *ColetorEstatisticas* são:

- Obter medidas descritivas, para cada atributo.
- Construir as distribuições de frequências de cada atributo.
- Calcular a média parcial, ou seja, média de um subconjunto de valores do atributo, esse subconjunto de valores pode ser filtrado a partir de valores de outros atributos.
- Identificar *outliers* pelas técnicas do teste *z-score* e diagrama de caixa.

3.3.2.7 Classe *GeradorGráficos*

Classe responsável por gerar as informações necessárias para realizar a plotagem das representações gráficas apresentadas na Seção 2.1.4.

A função privada *posicaoNoEixo* retorna qual a posição do valor em um eixo qualquer, esta será o próprio valor caso o mesmo seja quantitativo, caso o valor seja qualitativo, a posição será referente ao índice da categoria.

```
1  class GeradorGraficos
2  {
3  public:
4      typedef vector<double> Coordenada;
5      typedef vector<Coordenada> Coordenadas;
6      typedef vector<int> Frequencias;
7
8      static void gerarDiagramaDispersao2v(...);
9      static void gerarDiagramaDispersao3v(...);
10     static void gerarGraficoBarrasQualitativo(...);
11     static void gerarHistogramaQuantContinuo(...);
12     static void gerarGraficoBarrasQuantDiscreto(...);
13     static void gerarGraficoMultBarras(...);
14
15 private:
16     static double posicaoNoEixo(...);
17 };
```

Listagem 3.12: Classe *GeradorGráficos*

3.4 INTERFACE GRÁFICA

O desenvolvimento da interface foi a última atividade realizada e dentre todas foi a que mais exigiu tempo. As Seções a seguir apresentarão as principais telas da ferramenta, são elas:

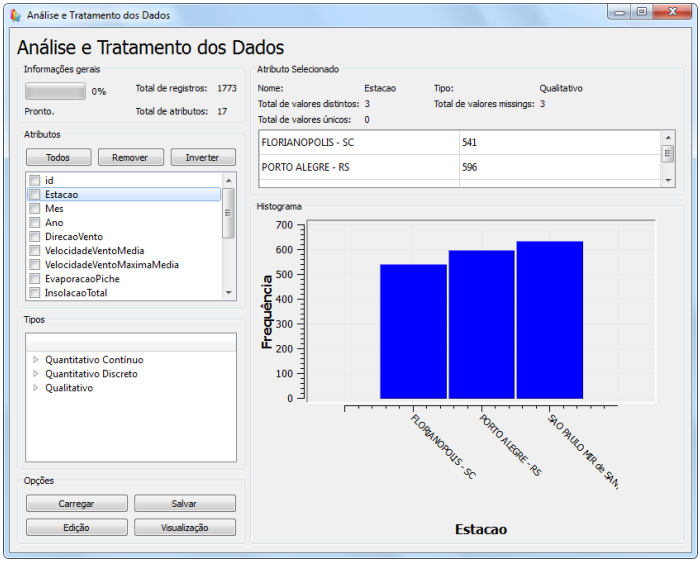
- Principal.
- Assistente de carregamento de arquivo.
- Edição.
- Visualização.

3.4.1 Tela principal

A tela principal, Figura 8, é semelhante ao Weka (Waikato Environment for Knowledge Analysis). Inicialmente nenhuma informação é exibida, já que nenhum dado foi carregado. Ao carregar algum dado, serão exibidas as seguintes informações:

- Gerais:
  - Total de atributos e registros.
  - Lista de atributos e seus tipos.
- Atributo selecionado
  - Tabela com medidas descritivas e a distribuição de frequências do atributo.
  - Total de valores *missings*, únicos e distintos. Para os atributos qualitativos ainda é mostrado o total de valores inválidos.
  - Histograma ou gráfico de barras dependendo do tipo do atributo.

Figura 8: Tela principal com dados



3.4.2 Tela assistente de carregamento

A tela de carregamento é composta por 8 etapas, de modo que as três primeiras são obrigatórias, enquanto as demais são opcionais e guiam o usuário através de uma pré-ATD. As etapas são:

1. Selecionar o arquivo que contenha os dados, Figura 9. O arquivo deve ser em formato CSV (Comma-separated values), contendo opcionalmente o nome dos atributos na primeira linha.

2. Escolher atributos (colunas) desejados, Figura 10.

3. Tipificar os atributos escolhidos, Figura 11.

4. Definir respectivamente quais intervalos, Figura 12, e categorias, Figura 13, os atributos quantitativos e qualitativos podem conter. Por padrão todos os valores são considerados.

5. Editar ou remover valores que estão fora dos intervalos ou categorias anteriormente definidos, Figura 14.

6. Escolher dentre duas técnicas, diagrama de caixa ou teste *z-score*, para identificar *outliers*, assim como definir seus parâmetros para cada atributo, Figura 15.

7. Editar, substituir ou remover valores que são considerados *outliers*, Figura 16.

8. Editar, substituir ou remover valores *missings* e inválidos, Figura 17. Nesta etapa também utilizar um atributo como referência e gerar novos registros de acordo com os valores faltantes do atributo.

Figura 9: Tela selecionar arquivo

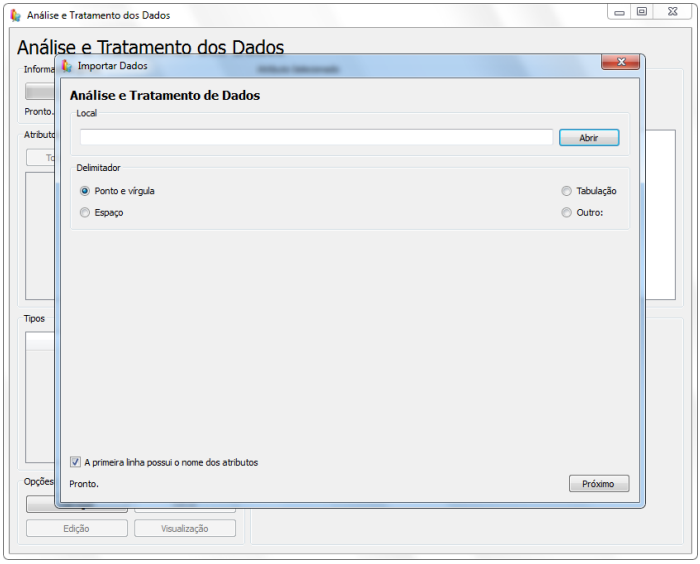


Figura 10: Tela escolher atributos

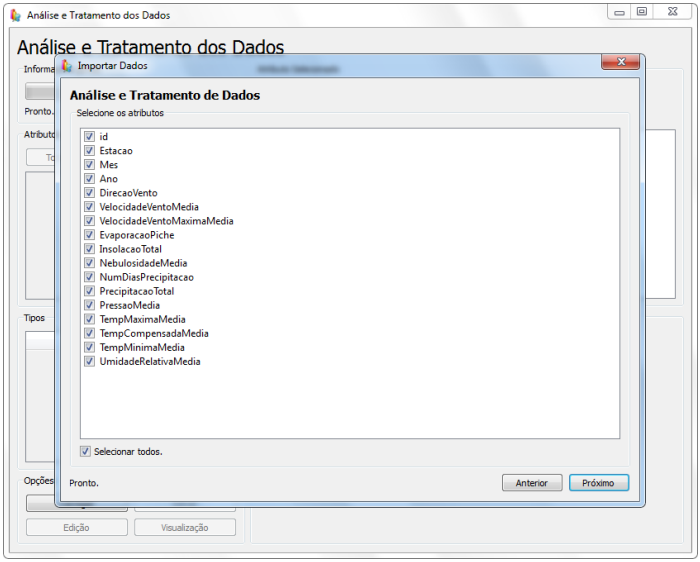


Figura 11: Tela tipificar atributos

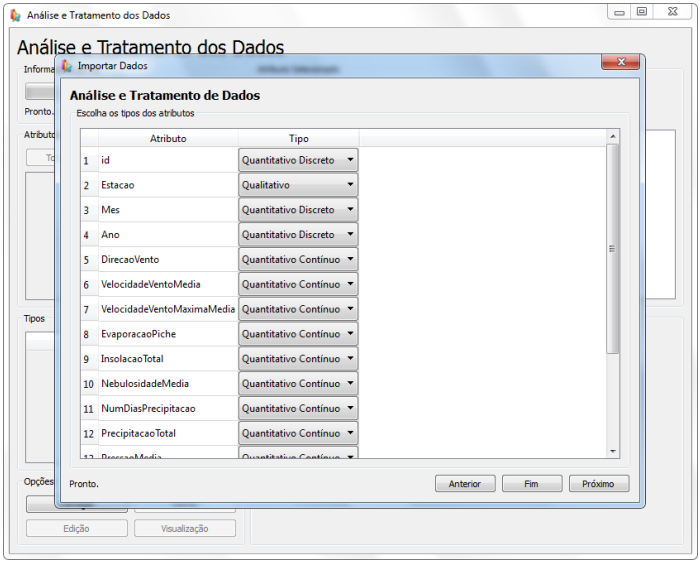


Figura 12: Tela definir intervalos quantitativos

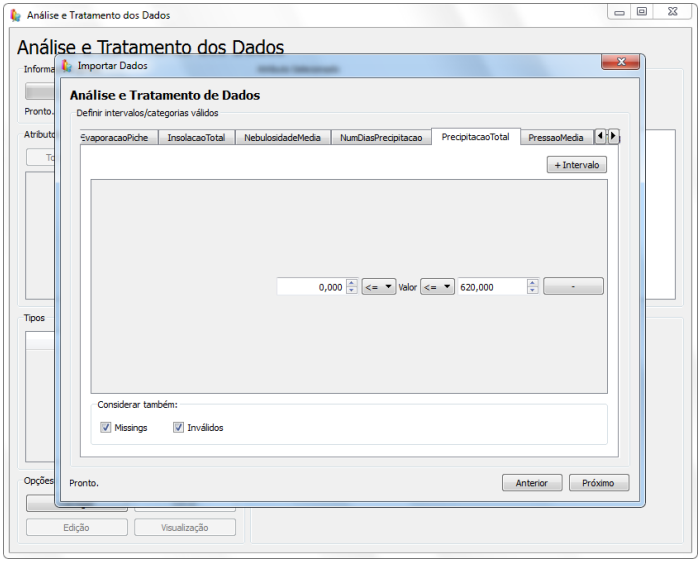


Figura 13: Tela definir categorias qualitativas

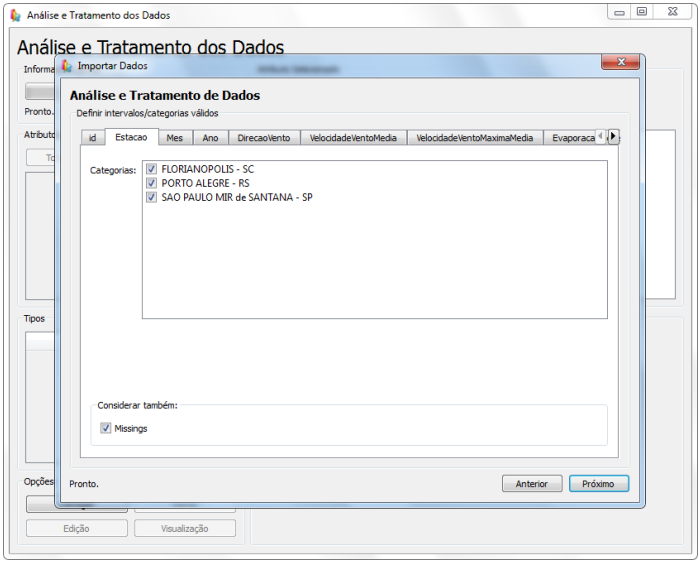


Figura 14: Tela editar valores fora dos intervalos ou categorias definidos

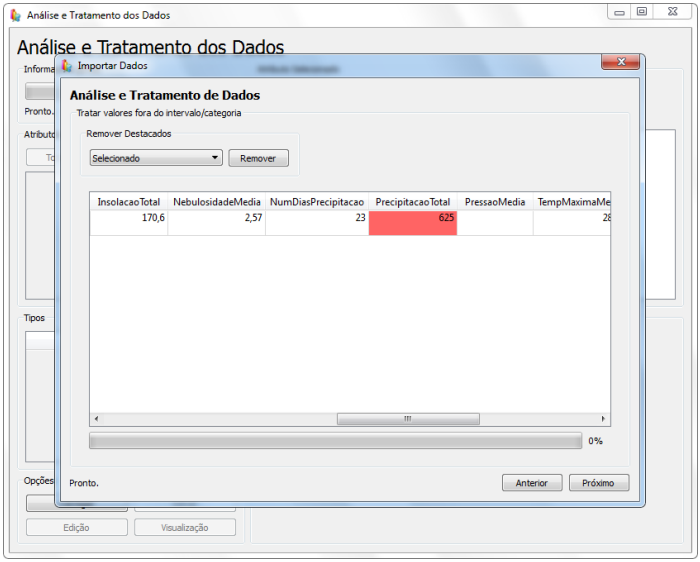


Figura 15: Tela escolher técnicas para identificar *outliers*

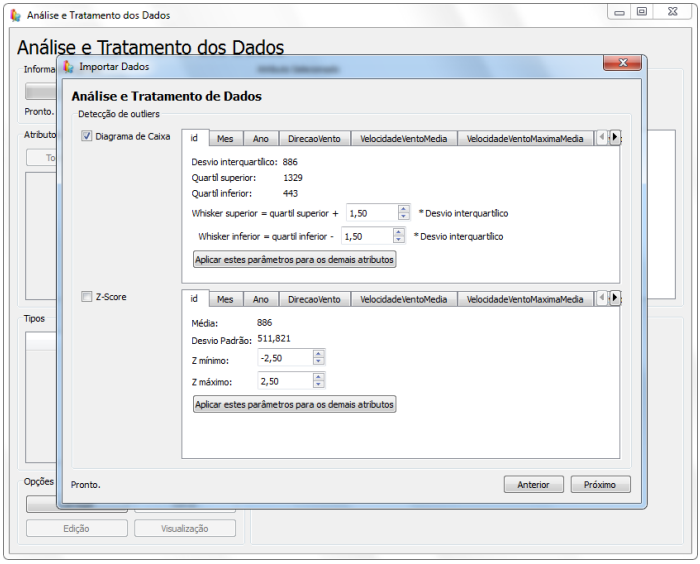


Figura 16: Tela editar valores *outliers*

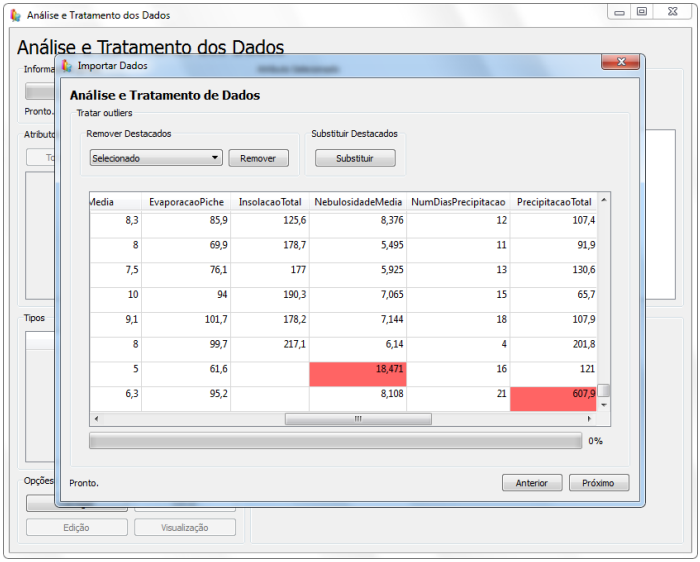
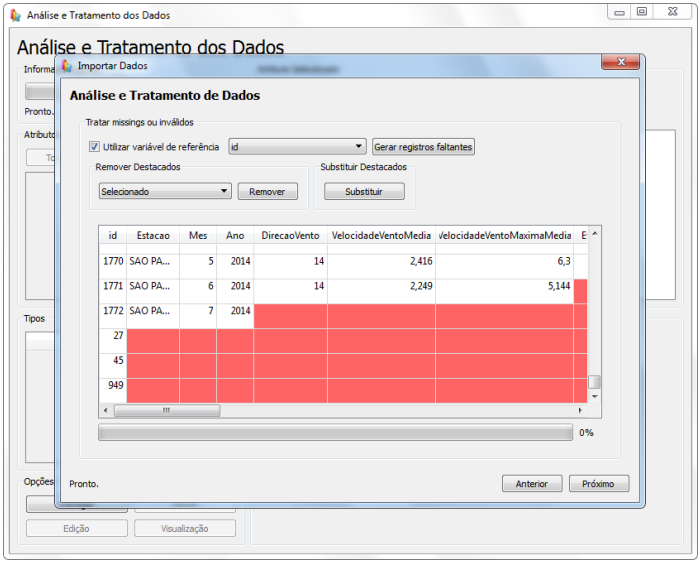


Figura 17: Tela editar *missings* e inválidos



3.4.3 Tela de visualização dos dados

Tela que contém as representações gráficas apresentadas na Seção 2.1.4, são elas:

- Diagrama de dispersão com dois e três atributos, Figuras 18 e 19 respectivamente.

- Gráfico de barras empilhadas, Figura 20.

- Diagrama de caixa, Figura 21.

Figura 18: Tela diagrama de dispersão com dois atributos.

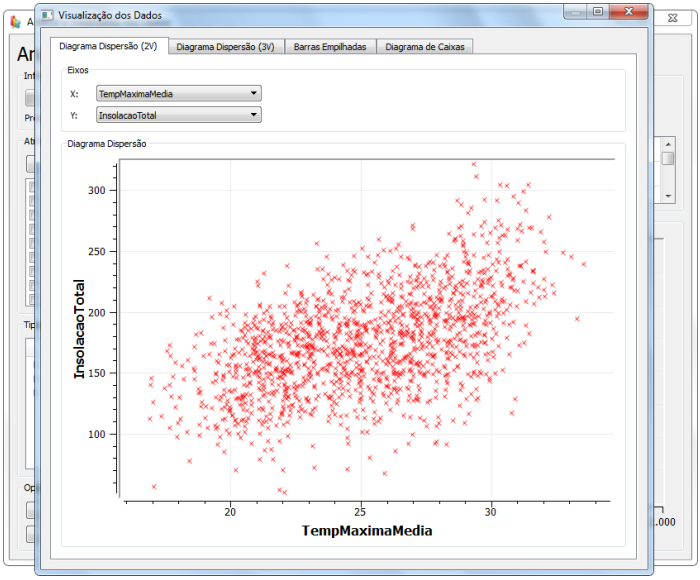


Figura 19: Tela diagrama de dispersão com três atributos

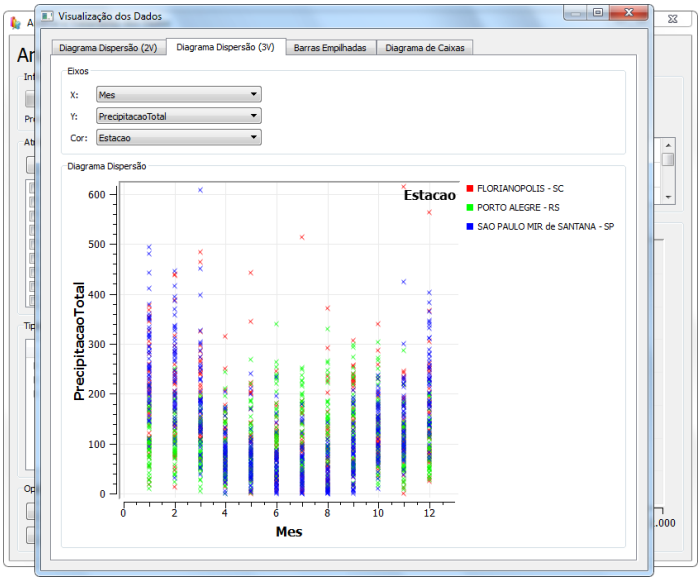


Figura 20: Tela gráfico de barras

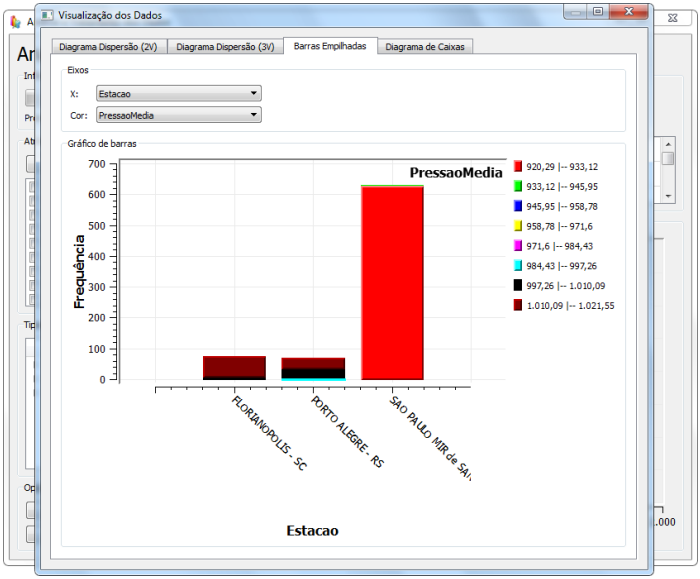
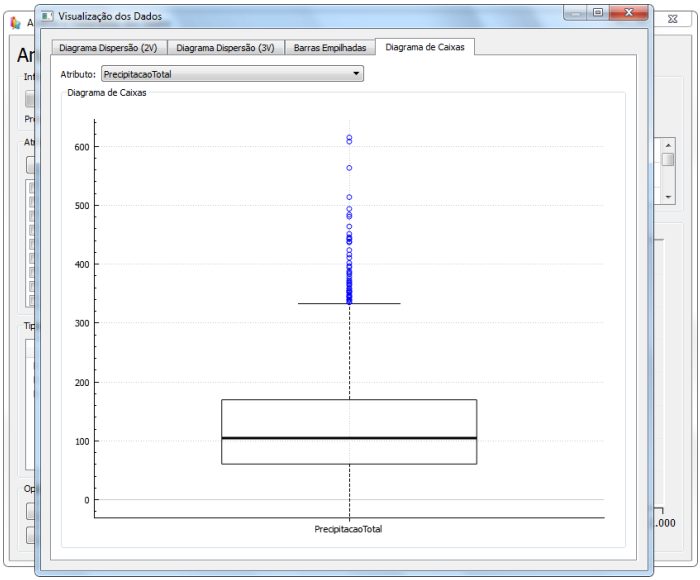


Figura 21: Tela diagrama de caixa



3.4.4 Tela de edição dos dados

A tela de edição, Figura 22, possibilita ao usuário manualmente editar os valores dos registros, além de fornecer um conjunto de funcionalidades de apoio, são elas:

- Remover ou adicionar atributos (colunas).
- Remover ou identificar registros por categoria ou intervalo de acordo com o tipo do atributo, Figuras 23 e 24
- Substituir um conjunto de valores qualitativos por um valor que já existe ou um novo, Figura 25.
- Substituir um conjunto de valores quantitativos por um valor específico, média ou média parcial. Também é possível construir DPs ou RLSs para preencherem os valores a serem substituídos. Figura 26.
- Preencher valor, atributo ou registro com DPs e RLSs, Figuras 27 e 28.
- Identificar *outliers* pelo diagrama de caixa ou teste *z-score*, Figuras 29 e 30 respectivamente.

Figura 22: Tela edição dos dados

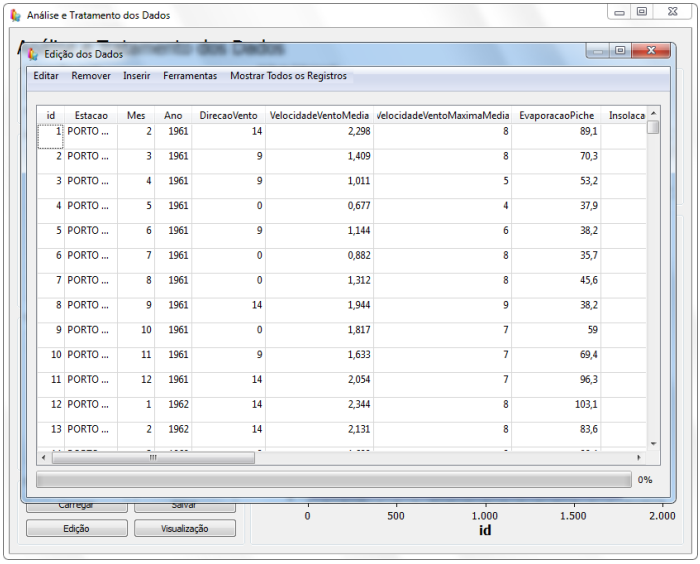


Figura 23: Tela remover ou identificar valores qualitativos

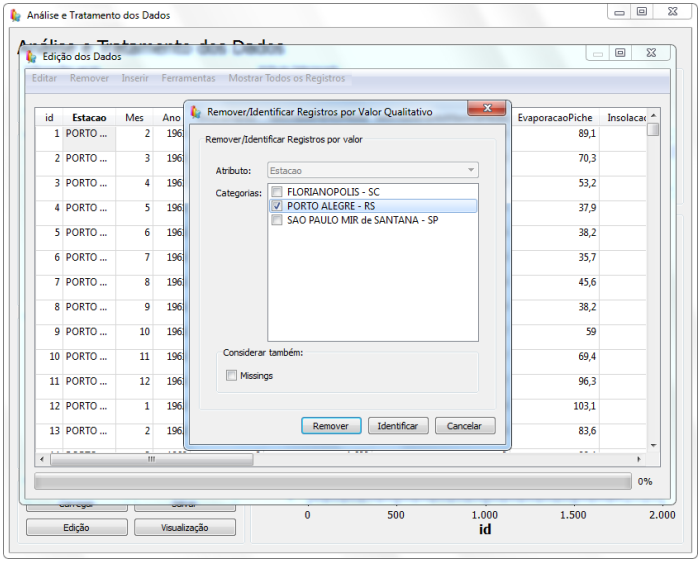


Figura 24: Tela remover ou identificar valores quantitativos

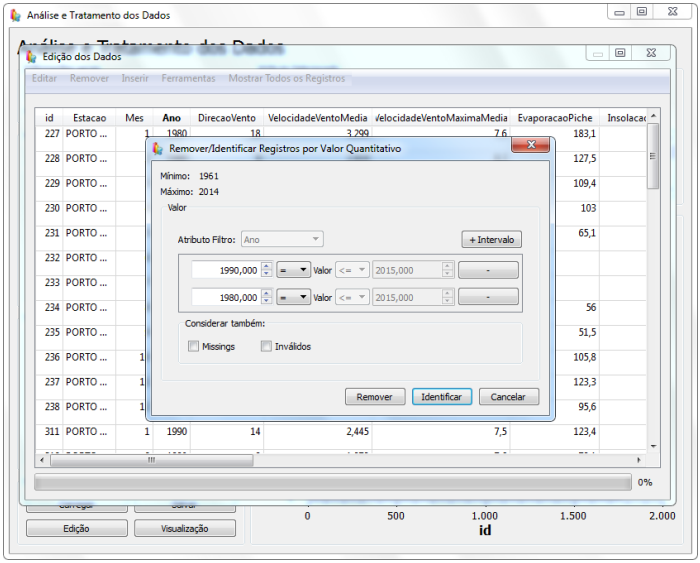


Figura 25: Tela substituir valores qualitativos

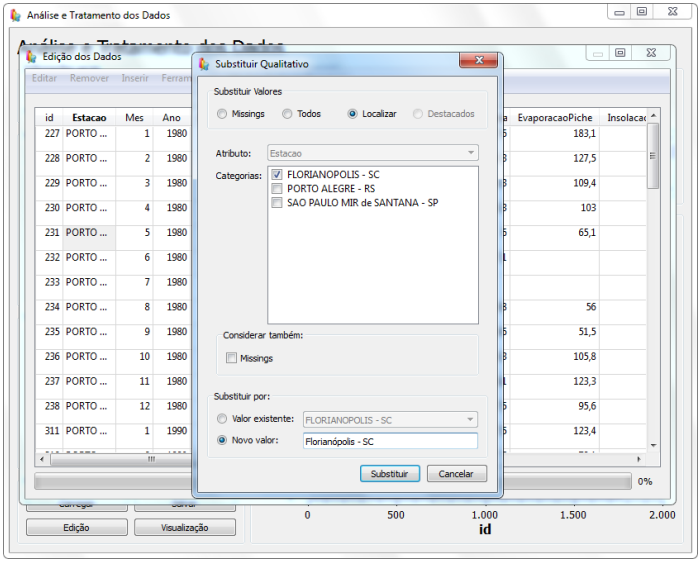


Figura 26: Tela substituir valores quantitativos

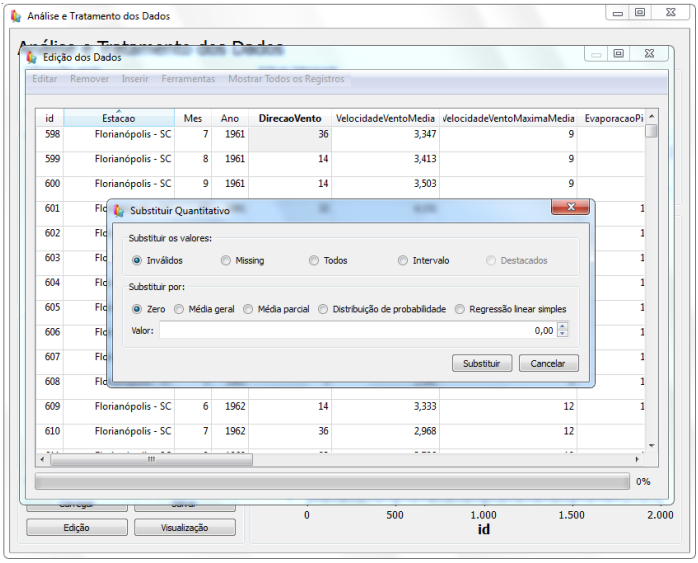


Figura 27: Tela gerar valor de uma DP

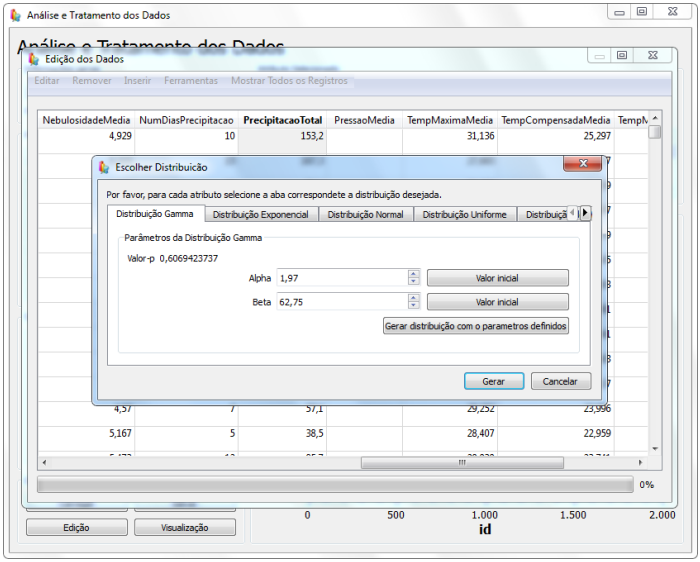


Figura 28: Tela gerar valor de uma RLS

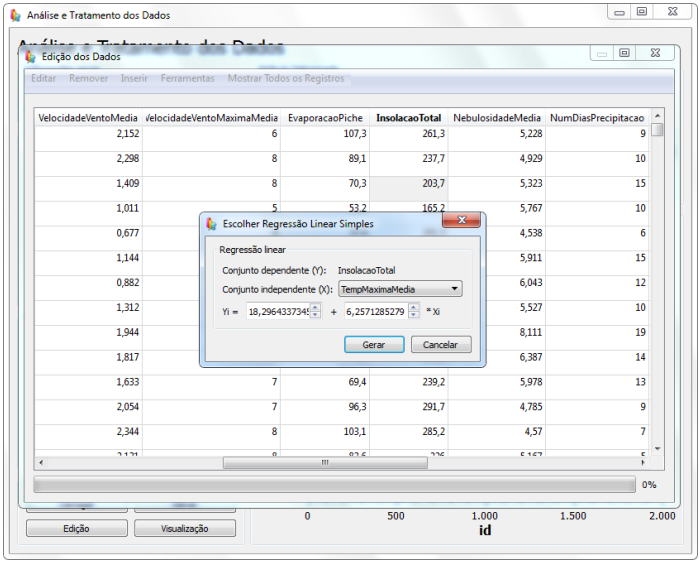


Figura 29: Tela identificar *outliers* pelo diagrama de caixa

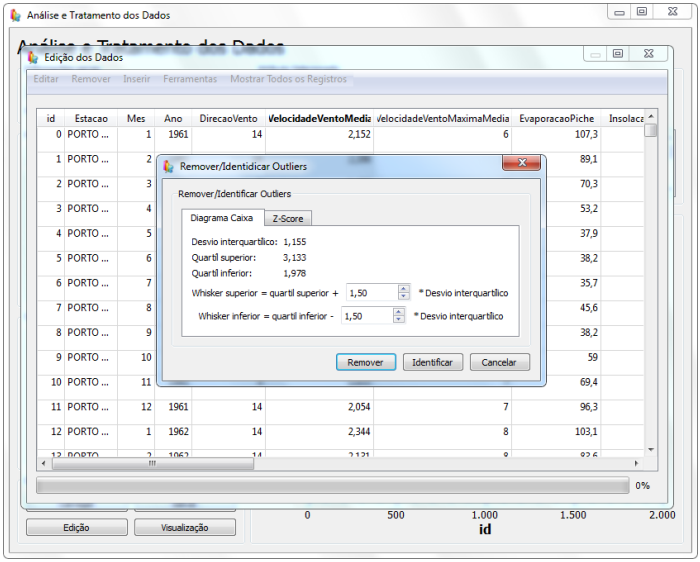
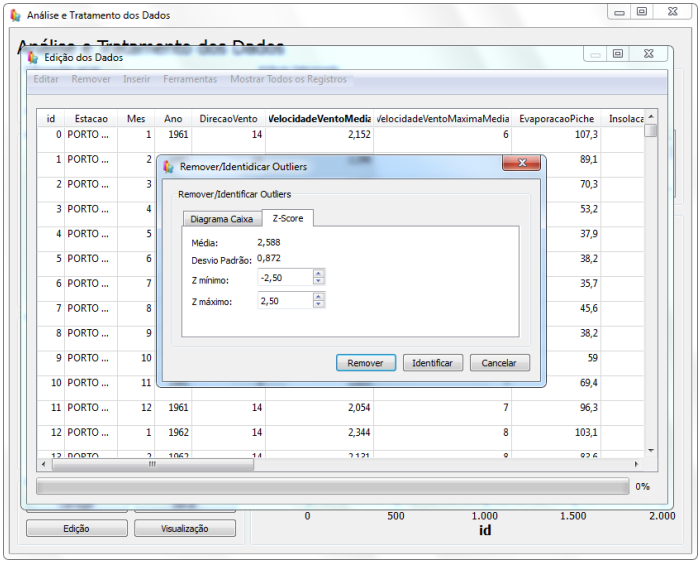


Figura 30: Tela identificar *outliers* pelo teste *z-score*



4 CONCLUSÃO

A motivação para a desenvolver esta ferramenta surgiu da necessidade de sua aplicação na área de previsão, em um dos projetos na qual o autor, orientador e membros da banca fazem parte. Contudo, a ATD é útil em qualquer área no momento que se deseja extrair informações e tratar um conjunto de dados.

Embora atualmente já exista inúmeras ferramentas que possuem mais recursos para ATD, como Weka, Excel e o SEstatNet (NASSAR; WRONSCKI; OHIRA, 2014), que já existem há mais de dez anos e ainda hoje veem sendo melhorados, a ferramenta desenvolvida se diferenciou por ser facilmente integrada ao projeto SO-BR.

A experiência de implementar vários conceitos estatísticos foi bastante enriquecedora. Além disso, fez aumentar ainda mais meus conhecimentos em programação além de muitas outras áreas da Ciência da Computação.

4.1 ALTERAÇÕES

A seguir são demonstradas as principais alterações entre o módulo de ATD presente no E&P Risk e a ferramenta desenvolvida neste trabalho.

- Acrescentado:
  - Medidas descritivas: amplitude, coeficiente de variação, quartis, desvio interquartílico e total de valores inválidos em um atributo quantitativo
  - Diagrama de dispersão com 3 atributos.
  - Diagrama de caixa entre as representações gráficas
  - No módulo de ATD do E&P Risk é possível substituir um valor por outro qualquer ou a média aritmética da população. Agora é possível substituir utilizando média parcial da população, ou gerando valor a partir de DPs e RLSs.
  - Sequência de passos que guiam o usuário através de uma pré-ATD.
  - Identificar/remover registros por filtros.
  - No módulo de ATD do E&P Risk já contém uma função para identificar valores *missings*, entretanto foram incluídas

funções para detectar registros *missings*, valores *outliers* e inválidos.
  - Adicionar novas categorias em atributos qualitativos

- Removido:
  - No módulo de ATD do E&P Risk os atributos qualitativos são classificados em ordinal e nominal. O motivo da retirada se deu pelo contexto do SO-BR não englobar essa subclassificação.
  - A tipificação automática presente no E&P Risk foi removida, pois no SO-BR os dados já veem tipificados do banco de dados, assim se o usuário carregar um arquivo ele terá que manualmente tipificar.

- Como já apresentado no Parágrafo 2.1.3.2.2, o número máximo de classes (k) na distribuição de frequências para atributos quantitativos contínuos passou a ser 25.

- Em termos de implementação a classe *ValorBase* e as que estendem dela foram criadas de afim de melhorar o desempenho, já que no módulo de ATD do E&P Risk todos os valores eram armazenados em strings (tipo texto), qualquer operação com atributos quantitativos era necessário converter computacionalmente de string para double (tipo numérico), que em termos de processamento é custoso. A classe *ValorQuantitativo* é uma extensão da classe *ValorBase* e já tem um atributo de classe que armazena o valor como um tipo numérico, dessa forma não há necessidade de realizar conversões.

- A interface gráfica foi toda refeita utilizando Qt Framework.

Além dessas muitas outras funcionalidades menores foram adicionadas.

4.2 TRABALHOS FUTUROS

Foram alguns os conceitos de ATD implementados nesta ferramenta, outras formas poderiam ter sido implementadas, mas devido ao curto prazo não foram. Entretanto, como este trabalho estará disponível para toda a sociedade, outros desenvolvedores fornecer melhorias ao código. Há inúmeros outros conceitos que podem ser incorporados a esta ferramenta.

As principais funcionalidades que poderiam ser acrescentadas:

- Ampliar a lib_fit de modo a suportar um maior número de distribuições de probabilidades, como por exemplo as distribuições discretas.
- Outras formas de representações gráficas, como por exemplo:
  - Diagrama de caixa em forma de distribuição.
  - Diagrama de ramo e folhas.
  - Gráfico de setores.
  - Gráficos tridimensionais.
- Mais técnicas de preenchimento/estimação de valores para os atributos correlacionadas além das RLSs, como redes neurais, árvores de decisão (ID3), etc.
- Medidas descritivas que indiquem a correlação entre atributos.
- Técnicas de detecção de *outliers* levando em consideração mais de um atributo, por exemplo utilizando clusterização (agrupamentos).

REFERÊNCIAS

BARBETTA, P. A.; REIS, M. M.; BORNIA, A. C. **Estatística: para cursos de engenharia e informática**. 3. ed. São Paulo: Atlas, 2010.

CHAMBERS, J. M. **Graphical methods for data analysis**. United Kingdom: Chapman and Hall/CRC, 1983.

FORMIGHIERI, S. **Uma biblioteca de funções estatísticas para o apoio no desenvolvimento de aplicações com aderência de dados**. 2007.

FREITAS FILHO, P. J. de. **Introdução à modelagem e simulação de sistemas**: Com aplicações em arena. 2. ed. Florianópolis: Visual Books, 2008.

GANTZ, J. F.; REINSEL, D. The expanding digital universe: A forecast of worldwide information growth through 2010. In: IDC. [S.l.], 2007.

HINES, W. W.; MONTGOMERY, D. C.; BORROR, D. M. G. C. M. **Probability and statistics in engineering**. New Jersey: John Wiley & Sons, 2008.

LAW, A. M.; KELTON, W. D.; KELTON, W. D. **Simulation modeling and analysis**. New York: McGraw-Hill, 1991.

LOHR, S. The age of big data. **New York Times**, 2012.

MEDRI, W. Análise exploratória de dados. v. 15, p. 05–13, 2011. Disponível em: <`http://www.uel.br/pos/estatisticaeducacao/textos\_didaticos/especializacao\_estatistica.pdf`>. Acesso em: 11 nov. 2014.

NASSAR, S. M.; WRONSCKI, V. R.; OHIRA, M. **SEstatNet**: Sistema especialista para o ensino de estatística na web. Florianópolis: UFSC, 2014. Disponível em: <http://www.sestat.net>. Acesso em: 10 nov. de 2014.

REIS, E. **Estatística descritiva**. 7. ed. Lisboa: Sílabo, 2008.

SANTIN, D. W. **Integração de uma Etapa de Análise Exploratória de Dados em um Sistema de Análise de Risco para Operações de Completação e Perfuração de Poços de Petróleo**. 2007.

SCOTT, D. W. **Multivariate density estimation: theory, practice, and visualization**. New Jersey: John Wiley & Sons, 2009.

SILVA, M. D. da. **Um modelo para análise e tratamento de dados de demanda de energia elétrica**. 2011.

STURGES, H. A. The choice of a class interval. **Journal of the American Statistical Association**, v. 21, n. 153, p. 65–66, 1926.

TODESCO, J. L. **Mineração de Dados**: Análise exploratória de dados. Florianópolis: UFSC, 2014. Disponível em: <`https://moodle.ufsc.br/`>. Acesso em: 11 nov. de 2014.

ANEXO A – Artigo

Desenvolvimento de uma Ferramenta para Análise e Tratamento de Dados

Rodolfo W. Reitz

Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brasil

rodolforeitz@gmail.com.br

Abstract. *This work involves a study of techniques for data analyzing and treatment, followed by the development of a tool that encompasses these techniques. The developed tool contains in terms of exploratory data analysis: descriptive measurements, frequency distributions, graphs, detection of abnormal values (outliers, invalids and missings). While in terms of data treatment, this tool has functionalities that help to improve the data, as fill or estimate values using average linear regressions or probability distributions. This tool is intended to be integrated into one of the projects that the author of this work participates.*

Resumo. *Este trabalho envolve um estudo de técnicas para análise e tratamento de dados, seguido pelo desenvolvimento de uma ferramenta que engloba estas técnicas. A ferramenta desenvolvida contém em termos de análise exploratória de dados: medidas descritivas, distribuições de frequência, gráficos, detecção de valores anormais (outliers, inválidos e missings). Enquanto em termos de tratamento de dados, esta ferramenta tem funcionalidades que ajudam a melhorar os dados, como preencher ou estimar valores usando média, regressões lineares ou distribuições de probabilidade. Esta ferramenta teve como finalidade ser integrada a um dos projetos que o autor deste trabalho participa.*

1. Introdução

Estima-se que a quantidade total de informações digitais presente no universo entre os anos de 2006 a 2010 aumentaria de 161 para 988 *exabytes* [Gantz e Reinsel 2007], um crescimento esperado anualmente de 57%. Este massivo incremento é um dos principais aspectos tratados pelo Big Data e sua causa está fortemente relacionada com a popularização de equipamentos de captura imagem, vídeo e áudio. Entretanto, uma outro motivo se deve ao fato que atualmente a maioria dos sistemas, como as cidades inteligentes ou redes de computadores, têm consigo sensores que monitoram ao longo do tempo variáveis referentes ao sistema em questão [Lohr 2012], de modo a comunicar as medições capturadas para algo que irá utilizá-la para tomar decisões necessárias para que o sistema alcance seu objetivo.

Tendo em vista esta crescente quantidade de informações, é cada vez maior a probabilidade da ocorrência de erros durante a sua captura, promovendo a redução da

qualidade dos dados, de modo que eles não representem corretamente a realidade. Os erros mais comuns são [Silva 2011]:

- De medições.
- De transcrição.
- Diferentes formatos de representação.
- Heterogeneidade, por exemplo, quando múltiplos sensores desigualmente calibrados monitoram uma mesma variável.
- Por algum motivo o valor a ser medido não estava disponível no momento.

Uma forma de tentar encontrar estes problemas é realizando uma análise exploratória dos dados (AED), que consiste na coleta, apresentação, análise e interpretação dos dados através de instrumentos adequados: tabelas, gráficos e indicadores. Já o tratamento dos dados (TD) consiste em adequar os problemas identificados de modo a melhor condizer com o esperado. Ambas, AED e TD juntas podem ser chamadas de análise e tratamento dos dados (ATD) [Reis 2008].

1.1. Importância

A ATD é de grande importância para as mais diversas aplicações que são baseadas em dados de entradas, dois exemplos serão listados a seguir:

- Aprendizado de máquina: Área da inteligência artificial dedicada na construção de modelos computacionais que possibilitam o computador aprender (extrair regras e padrões) a partir dos dados e utilizá-lo, por exemplo, para realizar previsões.
- Simulação de sistemas: Técnica utilizada com objetivo de imitar ou simular as operações de um processo em questão (sistema). A maioria dos sistemas possuem comportamento estocástico, afim de fazer com que a simulação possua o mesmo comportamento são utilizadas, por exemplo, distribuições de probabilidades para representar cada variável aleatória do sistema real, sendo que estas distribuições têm seus parâmetros obtidos através especialistas ou estimadores, contudo ambos se apoiam em dados do sistema real.

Essas são duas de muitas aplicação onde a ATD é essencial para o seu funcionamento, ambas são fortemente dirigidas por dados, de modo que a qualidade dos resultados estão diretamente relacionados com a qualidade da entrada. A qualidade da entrada determina a confiança de utilizar estes sistemas para apoio à tomadas de decisões.

1.2. Motivação

A principal motivação surgiu de um dos projetos que o autor deste trabalho participa durante a graduação junto ao PerformanceLab, na qual utiliza modelos computacionais para prever e otimizar o processo de perfuração de petróleo, titulado SO-BR (Sistema de Otimização de Brocas). Afim de buscar melhores resultados e dispor de algo para visualizar e editar os dados de entrada, surgiu a oportunidade de integrar ao projeto um módulo de ATD, que é a ferramenta desenvolvida neste trabalho.

Inicialmente o orientador e os membros da banca deste trabalho forneceram o código fonte de um projeto já desenvolvido pelo PerformanceLab, chamado de E&P Risk, na qual contém um módulo de ATD que foi codificado por [Santin 2007] em outro framework (C++ Builder) diferente do projeto SO-BR (Qt Framework). Este módulo de ATD do E&P Risk foi utilizado como apoio, porém foi amplamente melhorado em termos de codificação (*bugs*, otimizações), assim como novas funcionalidades foram adicionadas.

2. Trabalho

O trabalho consistiu em desenvolver de uma ferramenta que forneça suporte a ATD, que posteriormente foi integrada como um módulo no SO-BR. Mais especificamente as seguintes tarefas foram realizadas:

- Migrar e aperfeiçoar o módulo de ATD do E&P Risk desenvolvido em C++ Builder para Qt Framework.
- Investigar os métodos de AED: medidas descritivas e representações gráficas.
- Investigar estratégia de apoio ao TD, tais como preencher o valor inadequado utilizando distribuições de probabilidade ou regressões lineares.
- Codificar os conceitos estudados e desenvolver uma interface gráfica que permita utilizar as técnicas de ATD estudadas.

A linguagem de programação escolhida foi C++, na qual o autor deste trabalho tem maior experiência, além de apresentar um bom desempenho nos mais diversos problemas. Foram utilizadas as seguintes ferramentas de desenvolvimento:

- Microsoft Visual Studio 2010 Ultimate x86 (MVS): Ambiente de desenvolvimento (IDE).
- Qt 5.0.2 x86 MVS 2010 OpenGL: Ambiente de desenvolvimento (IDE).
- Qt Visual Studio Add-in 1.2.2: Integra ao MVS a maioria das funcionalidades disponibilizadas pelo Qt.

  As principais funcionalidades implementadas foram:

- Medidas descritivas
  - Total
    - Registros
    - Atributos
    - Valores inválidos e missings
    - Valores únicos e distintos de um atributo
  - Valores máximo e mínimo
  - Média
  - Amplitude
  - Desvio Padrão

    - Coeficiente de variação
    - Quartis (inferior, superior e mediana)
    - Desvio interquartílico
- Distribuição de frequências
- Representações gráficas
    - Histograma
    - Gráfico de barras
    - Diagrama de dispersão
    - Diagrama de caixa
- Detecção de possíveis registros *missings*
- Detecção de valores
    - *Missings*
    - Inválidos
    - *Outliers*
        - Diagrama de caixa
        - Teste *z-score*
    - Por filtro
- Tratamento dados
    - Remover/Inserir registros e atributos
    - Substituir valores identificados por filtro ou específicos por
        - Valor qualquer
        - Média da população ou de parte dela
        - Valores gerados a partir de distribuições de probabilidade ou regressões lineares simples
- Sequência de três passos para uma pré-ATD composta por
    - Definir respectivamente quais intervalos e categorias os atributos quantitativos e qualitativos podem conter. Por padrão todos os valores são considerados. Tratar os valores que não condizem aos intervalos e categorias definidas.
    - Detectar e tratar possíveis *outliers*.
    - Detectar e tratar valores ou registros *missings*.

3. Conclusão

A motivação para a desenvolver esta ferramenta surgiu da necessidade de sua aplicação na área de previsão, em um dos projetos na qual o autor, orientador e membros da banca

fazem parte. Contudo, a ATD é útil em qualquer área no momento que se deseja extrair informações e tratar um conjunto de dados.

Embora atualmente já exista inúmeras ferramentas que possuem mais recursos para ATD, como Weka, Excel e o SEstatNet [Nassar, Wronscki, Ohira et al. 2014], que já existem há mais de dez anos e ainda hoje veem sendo melhorados, a ferramenta desenvolvida se diferenciou por ser facilmente integrada ao projeto SO-BR.

A experiência de implementar vários conceitos estatísticos foi bastante enriquecedora. Além disso, fez aumentar ainda mais meus conhecimentos em programação além de muitas outras áreas da Ciência da Computação.

Referências

Gantz, J. F. e Reinsel, D. (2007). "The expanding digital universe". IDC.

Lohr, S. (2012). "The age of big data". New York Times.

Nassar, S. M., Wronscki, V. R., Ohira, M. at al (2014). "Sestatnet".

Reis, E. (2008). "Estatística descritiva". Sílabo, Lisboa, 7 edition.

Santin, D. W. (2007). "Integração de uma etapa de análise exploratória de dados em um sistema de análise de risco para operações de completação e perfuração de poços de Petróleo".

Silva, M. D. (2011). "Um modelo para análise e tratamento de dados de demanda de energia elétrica".

ANEXO B – Código fonte

```
1  ï»¿
2  Microsoft Visual Studio Solution File, Format Version 11.00
3  # Visual Studio 2010
4  Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "gui_analise_tratamento
       ", "AnaliseTratamento\AnaliseTratamento.vcxproj", "{8BFDB76B-FB12-4F8A
       -821B-C1B8F4B76E4C}"
5      ProjectSection(ProjectDependencies) = postProject
6          {55FB0313-A5AF-4418-A898-971E9BEE1FBC} = {55FB0313-A5AF-4418-A898
               -971E9BEE1FBC}
7          {2ABAEF46-6724-4740-9001-EE3D1760F5D7} = {2ABAEF46-6724-4740-9001-
               EE3D1760F5D7}
8          {E16D1D59-22F4-4120-947F-C78AC48714BD} = {E16D1D59-22F4-4120-947F-
               C78AC48714BD}
9          {7EF1B0C5-349A-4E03-B74C-CE33D6DC002F} = {7EF1B0C5-349A-4E03-B74C-
               CE33D6DC002F}
10         {7CC8E7FD-D035-45E8-AD6A-2E1B5DD9C0DC} = {7CC8E7FD-D035-45E8-AD6A-2
               E1B5DD9C0DC}
11     EndProjectSection
12 EndProject
13 Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "lib_qwt", "lib_qwt\
       lib_qwt.vcxproj", "{7EF1B0C5-349A-4E03-B74C-CE33D6DC002F}"
14 EndProject
15 Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "lib_analise_tratamento
       ", "lib_analise_tratamento\lib_analise_tratamento.vcxproj", "{E16D1D59
       -22F4-4120-947F-C78AC48714BD}"
16     ProjectSection(ProjectDependencies) = postProject
17         {2ABAEF46-6724-4740-9001-EE3D1760F5D7} = {2ABAEF46-6724-4740-9001-
               EE3D1760F5D7}
18         {7CC8E7FD-D035-45E8-AD6A-2E1B5DD9C0DC} = {7CC8E7FD-D035-45E8-AD6A-2
               E1B5DD9C0DC}
19     EndProjectSection
20 EndProject
21 Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "lib_utilidades", "
       lib_utilidades\lib_utilidades.vcxproj", "{2ABAEF46-6724-4740-9001-
       EE3D1760F5D7}"
22 EndProject
23 Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "lib_qcustomplot", "
       lib_qcustomplot\lib_qcustomplot.vcxproj", "{55FB0313-A5AF-4418-A898
       -971E9BEE1FBC}"
24 EndProject
25 Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "lib_fit", "lib_fit\
       lib_fit.vcxproj", "{7CC8E7FD-D035-45E8-AD6A-2E1B5DD9C0DC}"
26 EndProject
27 Global
28     GlobalSection(SolutionConfigurationPlatforms) = preSolution
29         Debug|Win32 = Debug|Win32
30         Release|Win32 = Release|Win32
31     EndGlobalSection
32     GlobalSection(ProjectConfigurationPlatforms) = postSolution
33         {8BFDB76B-FB12-4F8A-821B-C1B8F4B76E4C}.Debug|Win32.ActiveCfg =
               Debug|Win32
34         {8BFDB76B-FB12-4F8A-821B-C1B8F4B76E4C}.Debug|Win32.Build.0 = Debug|
               Win32
35         {8BFDB76B-FB12-4F8A-821B-C1B8F4B76E4C}.Release|Win32.ActiveCfg =
               Release|Win32
36         {8BFDB76B-FB12-4F8A-821B-C1B8F4B76E4C}.Release|Win32.Build.0 =
               Release|Win32
37         {7EF1B0C5-349A-4E03-B74C-CE33D6DC002F}.Debug|Win32.ActiveCfg =
               Debug|Win32
38         {7EF1B0C5-349A-4E03-B74C-CE33D6DC002F}.Debug|Win32.Build.0 = Debug|
               Win32
39         {7EF1B0C5-349A-4E03-B74C-CE33D6DC002F}.Release|Win32.ActiveCfg =
               Release|Win32
40         {7EF1B0C5-349A-4E03-B74C-CE33D6DC002F}.Release|Win32.Build.0 =
               Release|Win32
41         {E16D1D59-22F4-4120-947F-C78AC48714BD}.Debug|Win32.ActiveCfg =
               Debug|Win32
42         {E16D1D59-22F4-4120-947F-C78AC48714BD}.Debug|Win32.Build.0 = Debug|
               Win32
43         {E16D1D59-22F4-4120-947F-C78AC48714BD}.Release|Win32.ActiveCfg =
               Release|Win32
44         {E16D1D59-22F4-4120-947F-C78AC48714BD}.Release|Win32.Build.0 =
               Release|Win32
45         {2ABAEF46-6724-4740-9001-EE3D1760F5D7}.Debug|Win32.ActiveCfg =
               Debug|Win32
46         {2ABAEF46-6724-4740-9001-EE3D1760F5D7}.Debug|Win32.Build.0 = Debug|
               Win32
47         {2ABAEF46-6724-4740-9001-EE3D1760F5D7}.Release|Win32.ActiveCfg =
               Release|Win32
48         {2ABAEF46-6724-4740-9001-EE3D1760F5D7}.Release|Win32.Build.0 =
               Release|Win32
```

```
49          {55FB0313-A5AF-4418-A898-971E9BEE1FBC}.Debug|Win32.ActiveCfg =
                Debug|Win32
50          {55FB0313-A5AF-4418-A898-971E9BEE1FBC}.Debug|Win32.Build.0 = Debug|
                Win32
51          {55FB0313-A5AF-4418-A898-971E9BEE1FBC}.Release|Win32.ActiveCfg =
                Release|Win32
52          {55FB0313-A5AF-4418-A898-971E9BEE1FBC}.Release|Win32.Build.0 =
                Release|Win32
53          {7CC8E7FD-D035-45E8-AD6A-2E1B5DD9C0DC}.Debug|Win32.ActiveCfg =
                Debug|Win32
54          {7CC8E7FD-D035-45E8-AD6A-2E1B5DD9C0DC}.Debug|Win32.Build.0 = Debug|
                Win32
55          {7CC8E7FD-D035-45E8-AD6A-2E1B5DD9C0DC}.Release|Win32.ActiveCfg =
                Release|Win32
56          {7CC8E7FD-D035-45E8-AD6A-2E1B5DD9C0DC}.Release|Win32.Build.0 =
                Release|Win32
57      EndGlobalSection
58      GlobalSection(SolutionProperties) = preSolution
59          HideSolutionNode = FALSE
60      EndGlobalSection
61  EndGlobal
```

Listagem B.1: AnaliseTratamento.sln

B.1 SUBPROJETO GUI_ANALISE_TRATAMENTO

```
1  ï»¿<?xml version="1.0" encoding="utf-8"?>
2  <Project DefaultTargets="Build" ToolsVersion="4.0" xmlns="http://schemas.
       microsoft.com/developer/msbuild/2003">
3    <ItemGroup Label="ProjectConfigurations">
4      <ProjectConfiguration Include="Debug|Win32">
5        <Configuration>Debug</Configuration>
6        <Platform>Win32</Platform>
7      </ProjectConfiguration>
8      <ProjectConfiguration Include="Release|Win32">
9        <Configuration>Release</Configuration>
10       <Platform>Win32</Platform>
11     </ProjectConfiguration>
12   </ItemGroup>
13   <PropertyGroup Label="Globals">
14     <ProjectGuid>{8BFDB76B-FB12-4F8A-821B-C1B8F4B76E4C}</ProjectGuid>
15     <Keyword>Qt4VSv1.0</Keyword>
16     <ProjectName>gui_analise_tratamento</ProjectName>
17   </PropertyGroup>
18   <Import Project="$(VCTargetsPath)\Microsoft.Cpp.Default.props" />
19   <PropertyGroup Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'"
         Label="Configuration">
20     <ConfigurationType>Application</ConfigurationType>
21   </PropertyGroup>
22   <PropertyGroup Condition="'$(Configuration)|$(Platform)'=='Release|Win32'
         " Label="Configuration">
23     <ConfigurationType>Application</ConfigurationType>
24   </PropertyGroup>
25   <Import Project="$(VCTargetsPath)\Microsoft.Cpp.props" />
26   <ImportGroup Label="ExtensionSettings">
27   </ImportGroup>
28   <ImportGroup Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'"
         Label="PropertySheets">
29     <Import Project="$(UserRootDir)\Microsoft.Cpp.$(Platform).user.props"
           Condition="exists('$(UserRootDir)\Microsoft.Cpp.$(Platform).user.
           props')" Label="LocalAppDataPlatform" />
30   </ImportGroup>
31   <ImportGroup Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
         Label="PropertySheets">
32     <Import Project="$(UserRootDir)\Microsoft.Cpp.$(Platform).user.props"
           Condition="exists('$(UserRootDir)\Microsoft.Cpp.$(Platform).user.
           props')" Label="LocalAppDataPlatform" />
33   </ImportGroup>
34   <PropertyGroup Label="UserMacros" />
35   <PropertyGroup>
36     <_ProjectFileVersion>10.0.30319.1</_ProjectFileVersion>
37     <CodeAnalysisRuleSet Condition="'$(Configuration)|$(Platform)'=='Debug|
           Win32'">AllRules.ruleset</CodeAnalysisRuleSet>
```

```
38      <CodeAnalysisRules Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'" />
39      <CodeAnalysisRuleAssemblies Condition="'$(Configuration)|$(Platform)
            '=='Debug|Win32'" />
40      <CodeAnalysisRuleSet Condition="'$(Configuration)|$(Platform)'=='
            Release|Win32'">AllRules.ruleset</CodeAnalysisRuleSet>
41      <CodeAnalysisRules Condition="'$(Configuration)|$(Platform)'=='Release|
            Win32'" />
42      <CodeAnalysisRuleAssemblies Condition="'$(Configuration)|$(Platform)
            '=='Release|Win32'" />
43      <OutDir Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">$(
            SolutionDir)$(Platform)\$(Configuration)</OutDir>
44      <OutDir Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">$(
            SolutionDir)$(Platform)\$(Configuration)</OutDir>
45    </PropertyGroup>
46    <ItemDefinitionGroup Condition="'$(Configuration)|$(Platform)'=='Debug|
          Win32'">
47      <ClCompile>
48        <PreprocessorDefinitions>UNICODE;WIN32;QT_DLL;TCC_RODOLFO;QT_CORE_LIB
              ;QT_GUI_LIB;QT_WIDGETS_LIB;QT_PRINTSUPPORT_LIB;%(
              PreprocessorDefinitions)</PreprocessorDefinitions>
49        <AdditionalIncludeDirectories>.\GeneratedFiles;.\GeneratedFiles\$(
              ConfigurationName);.;$(QTDIR)\include;$(QTDIR)\include\QtCore;$(
              QTDIR)\include\QtGui;$(QTDIR)\include\QtWidgets;$(QTDIR)\include
              \QtPrintSupport;..\lib_analise_tratamento;..\lib_utilidades;..\
              lib_qwt;..\lib_qcustomplot;%(AdditionalIncludeDirectories)</
              AdditionalIncludeDirectories>
50        <Optimization>Disabled</Optimization>
51        <DebugInformationFormat>ProgramDatabase</DebugInformationFormat>
52        <RuntimeLibrary>MultiThreadedDebugDLL</RuntimeLibrary>
53        <TreatWChar_tAsBuiltInType>false</TreatWChar_tAsBuiltInType>
54      </ClCompile>
55      <Link>
56        <SubSystem>Windows</SubSystem>
57        <OutputFile>$(OutDir)\$(ProjectName).exe</OutputFile>
58        <AdditionalLibraryDirectories>$(QTDIR)\lib;$(SolutionDir)$(Platform)\
              $(Configuration);%(AdditionalLibraryDirectories)</
              AdditionalLibraryDirectories>
59        <GenerateDebugInformation>true</GenerateDebugInformation>
60        <AdditionalDependencies>qtmaind.lib;Qt5Cored.lib;Qt5Guid.lib;
              Qt5Widgetsd.lib;Qt5PrintSupportd.lib;lib_analise_tratamento.lib;
              lib_utilidades.lib;lib_qwt.lib;lib_qcustomplot.lib</
              AdditionalDependencies>
61        <Profile>false</Profile>
62      </Link>
63    </ItemDefinitionGroup>
64    <ItemDefinitionGroup Condition="'$(Configuration)|$(Platform)'=='Release|
          Win32'">
65      <ClCompile>
66        <PreprocessorDefinitions>UNICODE;WIN32;QT_DLL;TCC_RODOLFO;QT_NO_DEBUG
              ;NDEBUG;QT_CORE_LIB;QT_GUI_LIB;QT_WIDGETS_LIB;
              QT_PRINTSUPPORT_LIB;%(PreprocessorDefinitions)</
              PreprocessorDefinitions>
67        <AdditionalIncludeDirectories>.\GeneratedFiles;.\GeneratedFiles\$(
              ConfigurationName);.;$(QTDIR)\include;$(QTDIR)\include\QtCore;$(
              QTDIR)\include\QtGui;$(QTDIR)\include\QtWidgets;$(QTDIR)\include
              \QtPrintSupport;..\lib_analise_tratamento;..\lib_utilidades;..\
              lib_qwt;..\lib_qcustomplot;%(AdditionalIncludeDirectories)</
              AdditionalIncludeDirectories>
68        <DebugInformationFormat>
69        </DebugInformationFormat>
70        <RuntimeLibrary>MultiThreadedDLL</RuntimeLibrary>
71        <TreatWChar_tAsBuiltInType>false</TreatWChar_tAsBuiltInType>
72        <ShowIncludes>false</ShowIncludes>
73      </ClCompile>
74      <Link>
75        <SubSystem>Windows</SubSystem>
76        <OutputFile>$(OutDir)\$(ProjectName).exe</OutputFile>
77        <AdditionalLibraryDirectories>$(QTDIR)\lib;$(SolutionDir)$(Platform)\
              $(Configuration);%(AdditionalLibraryDirectories)</
              AdditionalLibraryDirectories>
78        <GenerateDebugInformation>false</GenerateDebugInformation>
79        <AdditionalDependencies>qtmain.lib;Qt5Core.lib;Qt5Gui.lib;Qt5Widgets.
              lib;Qt5PrintSupport.lib;lib_analise_tratamento.lib;
              lib_utilidades.lib;lib_qwt.lib;lib_qcustomplot.lib</
              AdditionalDependencies>
80        <Profile>false</Profile>
81      </Link>
82    </ItemDefinitionGroup>
83    <ItemGroup>
84      <ClCompile Include="diagrama_caixa.cpp" />
85      <ClCompile Include="diagrama_scatter.cpp" />
```

```
86      <ClCompile Include="dialog_escolher_RLS.cpp" />
87      <ClCompile Include="dialog_importacao_dados_pg_03_def_interv_cat.cpp"
            />
88      <ClCompile Include="dialog_importacao_dados_pg_00_arquivo.cpp" />
89      <ClCompile Include="dialog_importacao_dados_pg_01_selecionar_atribs.cpp
            " />
90      <ClCompile Include="dialog_importacao_dados_pg_05_tec_outliers.cpp" />
91      <ClCompile Include="dialog_importacao_dados_pg_02_tipificar.cpp" />
92      <ClCompile Include="
            dialog_importacao_dados_pg_04_tratar_fora_interv_cat.cpp" />
93      <ClCompile Include="
            dialog_importacao_dados_pg_07_tratar_missings_invalidos.cpp" />
94      <ClCompile Include="dialog_importacao_dados_pg_06_tratar_outliers.cpp"
            />
95      <ClCompile Include="GeneratedFiles\Debug\moc_dialog_edicao_dados.cpp">
96        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
97      </ClCompile>
98      <ClCompile Include="GeneratedFiles\Debug\
            moc_dialog_escolher_distribuicao.cpp">
99        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
100     </ClCompile>
101     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_escolher_RLS.cpp">
102       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
103     </ClCompile>
104     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_importacao_dados.
            cpp">
105       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
106     </ClCompile>
107     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_nova_coluna.cpp">
108       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
109     </ClCompile>
110     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_rem_ide_outliers.
            cpp">
111       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
112     </ClCompile>
113     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_rem_ide_quali.cpp">
114       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
115     </ClCompile>
116     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_rem_ide_quant.cpp">
117       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
118     </ClCompile>
119     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_renomear.cpp">
120       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
121     </ClCompile>
122     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_subst_qual.cpp">
123       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
124     </ClCompile>
125     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_subst_quant.cpp">
126       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
127     </ClCompile>
128     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_visualizar_graficos
            .cpp">
129       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
130     </ClCompile>
131     <ClCompile Include="GeneratedFiles\Debug\
            moc_grafico_barras_qual_quant_discreto.cpp">
132       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
133     </ClCompile>
134     <ClCompile Include="GeneratedFiles\Debug\moc_grafico_multibarras.cpp">
135       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
136     </ClCompile>
137     <ClCompile Include="GeneratedFiles\Debug\moc_histograma_quant_continuo.
            cpp">
138       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
139     </ClCompile>
140     <ClCompile Include="GeneratedFiles\Debug\moc_tela_inicial.cpp">
```

```
141       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
142     </ClCompile>
143     <ClCompile Include="GeneratedFiles\Debug\
            moc_thread_worker_coletar_estatisticas.cpp">
144       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
145     </ClCompile>
146     <ClCompile Include="GeneratedFiles\Debug\
            moc_widget_analise_exploracao_dados.cpp">
147       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
148     </ClCompile>
149     <ClCompile Include="GeneratedFiles\Debug\moc_widget_categorias_qual.cpp
            ">
150       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
151     </ClCompile>
152     <ClCompile Include="GeneratedFiles\Debug\moc_widget_distribuicao.cpp">
153       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
154     </ClCompile>
155     <ClCompile Include="GeneratedFiles\Debug\moc_widget_escolher_dp.cpp">
156       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
157     </ClCompile>
158     <ClCompile Include="GeneratedFiles\Debug\moc_widget_intervalos_quant.
            cpp">
159       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
160     </ClCompile>
161     <ClCompile Include="GeneratedFiles\Debug\moc_widget_intervalo_quant.cpp
            ">
162       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
163     </ClCompile>
164     <ClCompile Include="GeneratedFiles\Debug\moc_widget_parametro.cpp">
165       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
166     </ClCompile>
167     <ClCompile Include="GeneratedFiles\Debug\moc_widget_param_diag_caixa.
            cpp">
168       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
169     </ClCompile>
170     <ClCompile Include="GeneratedFiles\Debug\moc_widget_param_zscore.cpp">
171       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
172     </ClCompile>
173     <ClCompile Include="GeneratedFiles\Debug\
            moc_widget_regressao_linear_simples.cpp">
174       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
175     </ClCompile>
176     <ClCompile Include="GeneratedFiles\Debug\moc_widget_tabela_editora.cpp"
            >
177       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">true</ExcludedFromBuild>
178     </ClCompile>
179     <ClCompile Include="GeneratedFiles\qrc_tela_inicial.cpp">
180       <PrecompiledHeader Condition="'$(Configuration)|$(Platform)'==Debug|
              Win32'">
181       </PrecompiledHeader>
182       <PrecompiledHeader Condition="'$(Configuration)|$(Platform)'==
              Release|Win32'">
183       </PrecompiledHeader>
184     </ClCompile>
185     <ClCompile Include="GeneratedFiles\Release\moc_dialog_edicao_dados.cpp"
            >
186       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==Debug|
              Win32'">true</ExcludedFromBuild>
187     </ClCompile>
188     <ClCompile Include="GeneratedFiles\Release\
            moc_dialog_escolher_distribuicao.cpp">
189       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==Debug|
              Win32'">true</ExcludedFromBuild>
190     </ClCompile>
191     <ClCompile Include="GeneratedFiles\Release\moc_dialog_escolher_RLS.cpp"
            >
192       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'==Debug|
              Win32'">true</ExcludedFromBuild>
193     </ClCompile>
```

```
194     <ClCompile Include="GeneratedFiles\Release\moc_dialog_importacao_dados.
            cpp">
195       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
196     </ClCompile>
197     <ClCompile Include="GeneratedFiles\Release\moc_dialog_nova_coluna.cpp">
198       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
199     </ClCompile>
200     <ClCompile Include="GeneratedFiles\Release\moc_dialog_rem_ide_outliers.
            cpp">
201       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
202     </ClCompile>
203     <ClCompile Include="GeneratedFiles\Release\moc_dialog_rem_ide_quali.cpp
            ">
204       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
205     </ClCompile>
206     <ClCompile Include="GeneratedFiles\Release\moc_dialog_rem_ide_quant.cpp
            ">
207       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
208     </ClCompile>
209     <ClCompile Include="GeneratedFiles\Release\moc_dialog_renomear.cpp">
210       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
211     </ClCompile>
212     <ClCompile Include="GeneratedFiles\Release\moc_dialog_subst_qual.cpp">
213       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
214     </ClCompile>
215     <ClCompile Include="GeneratedFiles\Release\moc_dialog_subst_quant.cpp">
216       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
217     </ClCompile>
218     <ClCompile Include="GeneratedFiles\Release\
            moc_dialog_visualizar_graficos.cpp">
219       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
220     </ClCompile>
221     <ClCompile Include="GeneratedFiles\Release\
            moc_grafico_barras_qual_quant_discreto.cpp">
222       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
223     </ClCompile>
224     <ClCompile Include="GeneratedFiles\Release\moc_grafico_multibarras.cpp"
            >
225       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
226     </ClCompile>
227     <ClCompile Include="GeneratedFiles\Release\
            moc_histograma_quant_continuo.cpp">
228       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
229     </ClCompile>
230     <ClCompile Include="GeneratedFiles\Release\moc_tela_inicial.cpp">
231       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
232     </ClCompile>
233     <ClCompile Include="GeneratedFiles\Release\
            moc_thread_worker_coletar_estatisticas.cpp">
234       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
235     </ClCompile>
236     <ClCompile Include="GeneratedFiles\Release\
            moc_widget_analise_exploracao_dados.cpp">
237       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
238     </ClCompile>
239     <ClCompile Include="GeneratedFiles\Release\moc_widget_categorias_qual.
            cpp">
240       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
241     </ClCompile>
242     <ClCompile Include="GeneratedFiles\Release\moc_widget_distribuicao.cpp"
            >
243       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
244     </ClCompile>
245     <ClCompile Include="GeneratedFiles\Release\moc_widget_escolher_dp.cpp">
```

```
246         <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">true</ExcludedFromBuild>
247       </ClCompile>
248       <ClCompile Include="GeneratedFiles\Release\moc_widget_intervalos_quant.
              cpp">
249         <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">true</ExcludedFromBuild>
250       </ClCompile>
251       <ClCompile Include="GeneratedFiles\Release\moc_widget_intervalo_quant.
              cpp">
252         <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">true</ExcludedFromBuild>
253       </ClCompile>
254       <ClCompile Include="GeneratedFiles\Release\moc_widget_parametro.cpp">
255         <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">true</ExcludedFromBuild>
256       </ClCompile>
257       <ClCompile Include="GeneratedFiles\Release\moc_widget_param_diag_caixa.
              cpp">
258         <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">true</ExcludedFromBuild>
259       </ClCompile>
260       <ClCompile Include="GeneratedFiles\Release\moc_widget_param_zscore.cpp"
              >
261         <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">true</ExcludedFromBuild>
262       </ClCompile>
263       <ClCompile Include="GeneratedFiles\Release\
              moc_widget_regressao_linear_simples.cpp">
264         <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">true</ExcludedFromBuild>
265       </ClCompile>
266       <ClCompile Include="GeneratedFiles\Release\moc_widget_tabela_editora.
              cpp">
267         <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">true</ExcludedFromBuild>
268       </ClCompile>
269       <ClCompile Include="grafico_multibarras.cpp" />
270       <ClCompile Include="grafico_barras_qual_quant_discreto.cpp" />
271       <ClCompile Include="histograma_quant_continuo.cpp" />
272       <ClCompile Include="item_tabela_editora.cpp" />
273       <ClCompile Include="main.cpp" />
274       <ClCompile Include="populador.cpp" />
275       <ClCompile Include="tela_inicial.cpp" />
276       <ClCompile Include="dialog_rem_ide_outliers.cpp" />
277       <ClCompile Include="thread_worker_coletar_estatisticas.cpp" />
278       <ClCompile Include="dialog_nova_coluna.cpp" />
279       <ClCompile Include="dialog_edicao_dados.cpp" />
280       <ClCompile Include="dialog_rem_ide_quali.cpp" />
281       <ClCompile Include="dialog_rem_ide_quant.cpp" />
282       <ClCompile Include="dialog_renomear.cpp" />
283       <ClCompile Include="dialog_subst_qual.cpp" />
284       <ClCompile Include="dialog_subst_quant.cpp" />
285       <ClCompile Include="widget_analise_exploracao_dados.cpp" />
286       <ClCompile Include="dialog_importacao_dados.cpp" />
287       <ClCompile Include="dialog_visualizar_graficos.cpp" />
288       <ClCompile Include="widget_categorias_qual.cpp" />
289       <ClCompile Include="widget_distribuicao.cpp" />
290       <ClCompile Include="dialog_escolher_distribuicao.cpp" />
291       <ClCompile Include="widget_escolher_dp.cpp" />
292       <ClCompile Include="widget_intervalo_quant.cpp" />
293       <ClCompile Include="widget_intervalos_quant.cpp" />
294       <ClCompile Include="widget_parametro.cpp" />
295       <ClCompile Include="widget_param_diag_caixa.cpp" />
296       <ClCompile Include="widget_param_zscore.cpp" />
297       <ClCompile Include="widget_regressao_linear_simples.cpp" />
298       <ClCompile Include="widget_tabela_editora.cpp" />
299     </ItemGroup>
300     <ItemGroup>
301       <CustomBuild Include="tela_inicial.h">
302         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
303         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Moc%27ing tela_inicial.h...</Message>
304         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
                >
305         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
                DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
```

```
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
306       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
307       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing tela_inicial.h...</Message>
308       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
309       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
310     </CustomBuild>
311   </ItemGroup>
312   <ItemGroup>
313     <CustomBuild Include="tela_inicial.ui">
314       <FileType>Document</FileType>
315       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
            AdditionalInputs>
316       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Uic%27ing %(Identity)...</Message>
317       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
318       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
            FullPath)"</Command>
319       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
            AdditionalInputs>
320       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Uic%27ing %(Identity)...</Message>
321       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
322       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
            %(FullPath)"</Command>
323     </CustomBuild>
324   </ItemGroup>
325   <ItemGroup>
326     <ClInclude Include="diagrama_caixa.h" />
327     <ClInclude Include="diagrama_scatter.h" />
328     <CustomBuild Include="widget_tabela_editora.h">
329       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
330       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing widget_tabela_editora.h...</Message>
331       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
332       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
            DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
333       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
334       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing widget_tabela_editora.h...</Message>
335       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
336       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
```

```
ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
" "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
Program Files (x86)\Visual Leak Detector\include"</Command>
337     </CustomBuild>
338     <CustomBuild Include="dialog_escolher_RLS.h">
339       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
          Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
340       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
          Moc%27ing dialog_escolher_RLS.h...</Message>
341       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
          GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
          >
342       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
          (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
          ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
          DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
          DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
          .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
          include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
          "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
          QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
          lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
          Program Files (x86)\Visual Leak Detector\include"</Command>
343       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
          |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
344       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
          Moc%27ing dialog_escolher_RLS.h...</Message>
345       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
          >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
          Outputs>
346       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
          "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
          ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
          DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
          DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
          GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
          " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
          include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
          include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
          lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
          Program Files (x86)\Visual Leak Detector\include"</Command>
347     </CustomBuild>
348     <ClInclude Include="GeneratedFiles\ui_dialog_escolher_RLS.h" />
349     <ClInclude Include="GeneratedFiles\ui_widget_tabela_editora.h" />
350     <ClInclude Include="resource.h" />
351     <ClInclude Include="GeneratedFiles\ui_dialog_edicao_dados.h" />
352     <ClInclude Include="GeneratedFiles\ui_dialog_escolher_distribuicao.h"
          />
353     <ClInclude Include="GeneratedFiles\ui_dialog_importacao_dados.h" />
354     <ClInclude Include="GeneratedFiles\ui_dialog_nova_coluna.h" />
355     <ClInclude Include="GeneratedFiles\ui_dialog_rem_ide_outliers.h" />
356     <ClInclude Include="GeneratedFiles\ui_dialog_rem_ide_quali.h" />
357     <ClInclude Include="GeneratedFiles\ui_dialog_rem_ide_quant.h" />
358     <ClInclude Include="GeneratedFiles\ui_dialog_renomear.h" />
359     <ClInclude Include="GeneratedFiles\ui_dialog_subst_qual.h" />
360     <ClInclude Include="GeneratedFiles\ui_dialog_subst_quant.h" />
361     <ClInclude Include="GeneratedFiles\ui_dialog_visualizar_graficos.h" />
362     <CustomBuild Include="widget_regressao_linear_simples.h">
363       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
          Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
364       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
          Moc%27ing widget_regressao_linear_simples.h...</Message>
365       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
          GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
          >
366       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
          (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
          ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
          DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
          DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
          .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
          include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
          "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
          QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
          lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
          Program Files (x86)\Visual Leak Detector\include"</Command>
```

```
367         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
368         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Moc%27ing widget_regressao_linear_simples.h...</Message>
369         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
                Outputs>
370         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
                DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
                GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
                " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
                include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
                include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
371       </CustomBuild>
372       <CustomBuild Include="widget_param_zscore.h">
373         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
374         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Moc%27ing widget_param_zscore.h...</Message>
375         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
                >
376         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
                DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
                .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
                include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
                "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
                QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
377         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
378         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Moc%27ing widget_param_zscore.h...</Message>
379         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
                Outputs>
380         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
                DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
                GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
                " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
                include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
                include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
381       </CustomBuild>
382       <CustomBuild Include="widget_param_diag_caixa.h">
383         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
384         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Moc%27ing widget_param_diag_caixa.h...</Message>
385         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
                >
386         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
                DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
                .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
                include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
                "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
                QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
387         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
388         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Moc%27ing widget_param_diag_caixa.h...</Message>
389         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
```

```
            Outputs>
390       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
391     </CustomBuild>
392     <CustomBuild Include="dialog_rem_ide_outliers.h">
393       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
            FullPath)</AdditionalInputs>
394       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing dialog_rem_ide_outliers.h...</Message>
395       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
396       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
            DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
397       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
            ;%(FullPath)</AdditionalInputs>
398       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing dialog_rem_ide_outliers.h...</Message>
399       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
400       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
401     </CustomBuild>
402     <CustomBuild Include="widget_escolher_dp.h">
403       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
404       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing widget_escolher_dp.h...</Message>
405       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
406       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
            DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
407       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
408       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing widget_escolher_dp.h...</Message>
409       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
410       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
```

```
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
    </CustomBuild>
    <CustomBuild Include="widget_parametro.h">
      <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
      <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing widget_parametro.h...</Message>
      <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
      <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
            DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
      <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
      <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing widget_parametro.h...</Message>
      <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
      <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
    </CustomBuild>
    <CustomBuild Include="widget_distribuicao.h">
      <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
      <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing widget_distribuicao.h...</Message>
      <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
      <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
            DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
      <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
      <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing widget_distribuicao.h...</Message>
      <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
      <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
```

```
lib_utilidades" "-I.\..\lib_qwt" "-I.\..\lib_qcustomplot" "-IC:\
Program Files (x86)\Visual Leak Detector\include"</Command>
</CustomBuild>
<CustomBuild Include="dialog_escolher_distribuicao.h">
<AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
FullPath);$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
;%(FullPath)</AdditionalInputs>
<Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
Moc%27ing dialog_escolher_distribuicao.h...</Message>
<Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
>
<Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
.\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
"-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
QtPrintSupport" "-I.\..\lib_analise_tratamento" "-I.\..\
lib_utilidades" "-I.\..\lib_qwt" "-I.\..\lib_qcustomplot" "-IC:\
Program Files (x86)\Visual Leak Detector\include"</Command>
<AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
|Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
;%(FullPath);$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.
exe;%(FullPath)</AdditionalInputs>
<Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
Moc%27ing dialog_escolher_distribuicao.h...</Message>
<Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
>.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
Outputs>
<Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
"$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
" "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
include\QtPrintSupport" "-I.\..\lib_analise_tratamento" "-I.\..\
lib_utilidades" "-I.\..\lib_qwt" "-I.\..\lib_qcustomplot" "-IC:\
Program Files (x86)\Visual Leak Detector\include"</Command>
</CustomBuild>
<CustomBuild Include="widget_categorias_qual.h">
<AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
FullPath)</AdditionalInputs>
<Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
Moc%27ing widget_categorias_qual.h...</Message>
<Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
>
<Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
.\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
"-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
QtPrintSupport" "-I.\..\lib_analise_tratamento" "-I.\..\
lib_utilidades" "-I.\..\lib_qwt" "-I.\..\lib_qcustomplot" "-IC:\
Program Files (x86)\Visual Leak Detector\include"</Command>
<AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
|Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
;%(FullPath)</AdditionalInputs>
<Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
Moc%27ing widget_categorias_qual.h...</Message>
<Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
>.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
Outputs>
<Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
"$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
" "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
include\QtPrintSupport" "-I.\..\lib_analise_tratamento" "-I.\..\
```

```
            lib_utilidades" "-I.\..\lib_qwt" "-I.\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
451     </CustomBuild>
452     <CustomBuild Include="widget_intervalos_quant.h">
453       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
            FullPath);$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
454       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing widget_intervalos_quant.h...</Message>
455       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
456       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
            DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
457       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
            ;%(FullPath);$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
458       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing widget_intervalos_quant.h...</Message>
459       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
460       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
461     </CustomBuild>
462     <CustomBuild Include="widget_intervalo_quant.h">
463       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
            FullPath);$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
464       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing widget_intervalo_quant.h...</Message>
465       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
466       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
            DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
467       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
            ;%(FullPath);$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
468       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing widget_intervalo_quant.h...</Message>
469       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
470       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
```

```
471     </CustomBuild>
472     <CustomBuild Include="dialog_nova_coluna.h">
473       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
              FullPath)</AdditionalInputs>
474       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Moc%27ing dialog_nova_coluna.h...</Message>
475       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
              >
476       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
              DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
              .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
              include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
              "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
              QtPrintSupport" "-I..\lib_analise_tratamento" "-I..\
              lib_utilidades" "-I..\lib_qwt" "-I..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
477       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
              ;%(FullPath)</AdditionalInputs>
478       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Moc%27ing dialog_nova_coluna.h...</Message>
479       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
              Outputs>
480       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
              DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
              GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
              " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
              include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
              include\QtPrintSupport" "-I..\lib_analise_tratamento" "-I..\
              lib_utilidades" "-I..\lib_qwt" "-I..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
481     </CustomBuild>
482     <CustomBuild Include="dialog_importacao_dados.h">
483       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
              FullPath)</AdditionalInputs>
484       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Moc%27ing dialog_importacao_dados.h...</Message>
485       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
              >
486       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
              DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
              .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
              include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
              "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
              QtPrintSupport" "-I..\lib_analise_tratamento" "-I..\
              lib_utilidades" "-I..\lib_qwt" "-I..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
487       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
              ;%(FullPath)</AdditionalInputs>
488       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Moc%27ing dialog_importacao_dados.h...</Message>
489       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
              Outputs>
490       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
              DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
              GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
              " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
              include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
              include\QtPrintSupport" "-I..\lib_analise_tratamento" "-I..\
              lib_utilidades" "-I..\lib_qwt" "-I..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
491     </CustomBuild>
492     <CustomBuild Include="dialog_subst_quant.h">
```

```
493         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
                FullPath)</AdditionalInputs>
494         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Moc%27ing dialog_subst_quant.h...</Message>
495         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
                >
496         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
                DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
                .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
                include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
                "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
                QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
497         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
                ;%(FullPath)</AdditionalInputs>
498         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Moc%27ing dialog_subst_quant.h...</Message>
499         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
                Outputs>
500         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
                DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
                GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
                " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
                include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
                include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
501       </CustomBuild>
502       <CustomBuild Include="dialog_subst_qual.h">
503         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
                FullPath)</AdditionalInputs>
504         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Moc%27ing dialog_subst_qual.h...</Message>
505         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
                >
506         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
                DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
                .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
                include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
                "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
                QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
507         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
                ;%(FullPath)</AdditionalInputs>
508         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Moc%27ing dialog_subst_qual.h...</Message>
509         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
                Outputs>
510         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
                DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
                GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
                " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
                include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
                include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
511       </CustomBuild>
512       <CustomBuild Include="dialog_renomear.h">
513         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
```

```
            FullPath)</AdditionalInputs>
514         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing dialog_renomear.h...</Message>
515         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
516         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
            DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
517         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
            ;%(FullPath)</AdditionalInputs>
518         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing dialog_renomear.h...</Message>
519         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
520         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
521       </CustomBuild>
522       <CustomBuild Include="dialog_rem_ide_quant.h">
523         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
            FullPath)</AdditionalInputs>
524         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing dialog_rem_ide_quant.h...</Message>
525         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
526         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
            DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
            .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
            include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
            "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
            QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
527         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
            ;%(FullPath)</AdditionalInputs>
528         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing dialog_rem_ide_quant.h...</Message>
529         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
530         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
            GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
            " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
            include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
            include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
            lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
            Program Files (x86)\Visual Leak Detector\include"</Command>
531       </CustomBuild>
532       <CustomBuild Include="dialog_rem_ide_quali.h">
533         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
            FullPath)</AdditionalInputs>
```

```
534         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Moc%27ing dialog_rem_ide_quali.h...</Message>
535         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
                >
536         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
                DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
                .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
                include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
                "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
                QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
537         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
                ;%(FullPath)</AdditionalInputs>
538         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Moc%27ing dialog_rem_ide_quali.h...</Message>
539         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
                Outputs>
540         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
                DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
                GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
                " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
                include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
                include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
541       </CustomBuild>
542       <CustomBuild Include="grafico_multibarras.h">
543         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
                FullPath)</AdditionalInputs>
544         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Moc%27ing grafico_multibarras.h...</Message>
545         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
                >
546         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
                DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I.\
                GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\include
                " "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui" "-I$(
                QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\QtPrintSupport" "-
                I..\..\lib_analise_tratamento" "-I..\..\lib_utilidades" "-I..\..\
                lib_qwt" "-I..\..\lib_qcustomplot"</Command>
547         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
                ;%(FullPath)</AdditionalInputs>
548         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Moc%27ing grafico_multibarras.h...</Message>
549         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
                Outputs>
550         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
                DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
                GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
                " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
                include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
                include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot"</
                Command>
551       </CustomBuild>
552       <CustomBuild Include="dialog_visualizar_graficos.h">
553         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
                FullPath)</AdditionalInputs>
554         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Moc%27ing dialog_visualizar_graficos.h...</Message>
```

```
555       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
              >
556       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
              DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
              .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
              include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
              "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
              QtPrintSupport" "-I..\lib_analise_tratamento" "-I..\
              lib_utilidades" "-I..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
557       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
              ;%(FullPath)</AdditionalInputs>
558       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Moc%27ing dialog_visualizar_graficos.h...</Message>
559       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
              Outputs>
560       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
              DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
              GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
              " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
              include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
              include\QtPrintSupport" "-I..\lib_analise_tratamento" "-I..\..\
              lib_utilidades" "-I..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
561     </CustomBuild>
562     <CustomBuild Include="thread_worker_coletar_estatisticas.h">
563       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
564       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Moc%27ing thread_worker_coletar_estatisticas.h...</Message>
565       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
              >
566       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
              DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I
              .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
              include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
              "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
              QtPrintSupport" "-I..\lib_analise_tratamento" "-I..\..\
              lib_utilidades" "-I..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
567       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
568       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Moc%27ing thread_worker_coletar_estatisticas.h...</Message>
569       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
              Outputs>
570       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
              DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
              GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
              " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
              include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
              include\QtPrintSupport" "-I..\lib_analise_tratamento" "-I..\..\
              lib_utilidades" "-I..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
571     </CustomBuild>
572     <ClInclude Include="escala_nomes.h" />
573     <ClInclude Include="GeneratedFiles\ui_tela_inicial.h" />
574     <CustomBuild Include="histograma_quant_continuo.h">
575       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
              FullPath);$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
              ;%(FullPath);$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
576       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Moc%27ing histograma_quant_continuo.h...</Message>
```

```
577         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
                >
578         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
                DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
                .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
                include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
                "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
                QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
579         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
                ;%(FullPath);$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.
                exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(FullPath)</
                AdditionalInputs>
580         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Moc%27ing histograma_quant_continuo.h...</Message>
581         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
                Outputs>
582         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
                DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
                GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
                " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
                include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
                include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
583     </CustomBuild>
584     <CustomBuild Include="grafico_barras_qual_quant_discreto.h">
585       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
                FullPath);$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
                ;%(FullPath)</AdditionalInputs>
586         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Moc%27ing grafico_barras_qual_quant_discreto.h...</Message>
587         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
                >
588         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
                DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\GeneratedFiles" "-I.\
                GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\include
                " "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui" "-I$(
                QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\QtPrintSupport" "-
                I..\..\lib_analise_tratamento" "-I..\..\lib_utilidades" "-I..\..\
                lib_qwt" "-I..\..\lib_qcustomplot"</Command>
589         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
                ;%(FullPath);$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.
                exe;%(FullPath)</AdditionalInputs>
590         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Moc%27ing grafico_barras_qual_quant_discreto.h...</Message>
591         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
                Outputs>
592         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
                ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
                DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
                DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB "-I.\
                GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
                " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
                include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
                include\QtPrintSupport" "-I..\..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot"</
                Command>
593     </CustomBuild>
594     <ClInclude Include="GeneratedFiles\ui_widget_analise_exploracao_dados.h
            " />
595     <ClInclude Include="GeneratedFiles\ui_widget_categorias_qual.h" />
596     <ClInclude Include="GeneratedFiles\ui_widget_distribuicao.h" />
597     <ClInclude Include="GeneratedFiles\ui_widget_escolher_dp.h" />
```

```
598     <ClInclude Include="GeneratedFiles\ui_widget_intervalos_quant.h" />
599     <ClInclude Include="GeneratedFiles\ui_widget_intervalo_quant.h" />
600     <ClInclude Include="GeneratedFiles\ui_widget_parametro.h" />
601     <ClInclude Include="GeneratedFiles\ui_widget_param_diag_caixa.h" />
602     <ClInclude Include="GeneratedFiles\ui_widget_param_zscore.h" />
603     <ClInclude Include="GeneratedFiles\ui_widget_regressao_linear_simples.h
            " />
604     <ClInclude Include="item_tabela_editora.h" />
605     <ClInclude Include="populador.h" />
606     <CustomBuild Include="widget_analise_exploracao_dados.h">
607       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
              FullPath)</AdditionalInputs>
608       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Moc%27ing widget_analise_exploracao_dados.h...</Message>
609       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
              >
610       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
              DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
              .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
              include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
              "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
              QtPrintSupport" "-I.\..\lib_analise_tratamento" "-I.\..\
              lib_utilidades" "-I.\..\lib_qwt" "-I.\..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
611       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
              ;%(FullPath)</AdditionalInputs>
612       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Moc%27ing widget_analise_exploracao_dados.h...</Message>
613       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
              Outputs>
614       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
              DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
              GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
              " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
              include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
              include\QtPrintSupport" "-I.\..\lib_analise_tratamento" "-I.\..\
              lib_utilidades" "-I.\..\lib_qwt" "-I.\..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
615     </CustomBuild>
616     <CustomBuild Include="dialog_edicao_dados.h">
617       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe;%(
              FullPath);%(AdditionalInputs)</AdditionalInputs>
618       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Moc%27ing dialog_edicao_dados.h...</Message>
619       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
              >
620       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_CORE_LIB -DQT_GUI_LIB -
              DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\GeneratedFiles" "-I
              .\GeneratedFiles\$(ConfigurationName)\." "-I." "-I$(QTDIR)\
              include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtGui"
              "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\include\
              QtPrintSupport" "-I.\..\lib_analise_tratamento" "-I.\..\
              lib_utilidades" "-I.\..\lib_qwt" "-I.\..\lib_qcustomplot" "-IC:\
              Program Files (x86)\Visual Leak Detector\include"</Command>
621       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath);$(QTDIR)\bin\moc.exe
              ;%(FullPath);%(AdditionalInputs)</AdditionalInputs>
622       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Moc%27ing dialog_edicao_dados.h...</Message>
623       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
              Outputs>
624       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\moc.exe"  "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp"  -DUNICODE -DWIN32 -
              DQT_DLL -DTCC_RODOLFO -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
              DQT_GUI_LIB -DQT_WIDGETS_LIB -DQT_PRINTSUPPORT_LIB  "-I.\
```

```
                GeneratedFiles" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I.
                " "-I$(QTDIR)\include" "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\
                include\QtGui" "-I$(QTDIR)\include\QtWidgets" "-I$(QTDIR)\
                include\QtPrintSupport" "-I..\lib_analise_tratamento" "-I..\..\
                lib_utilidades" "-I..\..\lib_qwt" "-I..\..\lib_qcustomplot" "-IC:\
                Program Files (x86)\Visual Leak Detector\include"</Command>
625       </CustomBuild>
626     </ItemGroup>
627     <ItemGroup>
628       <CustomBuild Include="tela_inicial.qrc">
629         <FileType>Document</FileType>
630         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">%(FullPath);%(AdditionalInputs)</AdditionalInputs>
631         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Rcc%27ing %(Identity)...</Message>
632         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\qrc_%(Filename).cpp;%(Outputs)</Outputs>
633         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\rcc.exe" -name "%(Filename)" -no-compress "%(
                FullPath)" -o .\GeneratedFiles\qrc_%(Filename).cpp</Command>
634         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">%(FullPath);%(AdditionalInputs)</AdditionalInputs>
635         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Rcc%27ing %(Identity)...</Message>
636         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\qrc_%(Filename).cpp;%(Outputs)</Outputs>
637         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\rcc.exe" -name "%(Filename)" -no-compress "%(
                FullPath)" -o .\GeneratedFiles\qrc_%(Filename).cpp</Command>
638       </CustomBuild>
639     </ItemGroup>
640     <ItemGroup>
641       <CustomBuild Include="dialog_edicao_dados.ui">
642         <FileType>Document</FileType>
643         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
644         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
645         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
646         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
647         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
648         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
649         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
650         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
651       </CustomBuild>
652     </ItemGroup>
653     <ItemGroup>
654       <CustomBuild Include="widget_analise_exploracao_dados.ui">
655         <FileType>Document</FileType>
656         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
657         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
658         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
659         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
660         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
661         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
662         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
663         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
664       </CustomBuild>
665     </ItemGroup>
666     <ItemGroup>
```

```
667     <CustomBuild Include="dialog_visualizar_graficos.ui">
668       <FileType>Document</FileType>
669       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
670       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Uic%27ing %(Identity)...</Message>
671       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
672       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
              FullPath)"</Command>
673       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
674       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Uic%27ing %(Identity)...</Message>
675       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
676       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
              %(FullPath)"</Command>
677     </CustomBuild>
678   </ItemGroup>
679   <ItemGroup>
680     <CustomBuild Include="dialog_rem_ide_quali.ui">
681       <FileType>Document</FileType>
682       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
683       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Uic%27ing %(Identity)...</Message>
684       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
685       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
              FullPath)"</Command>
686       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
687       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Uic%27ing %(Identity)...</Message>
688       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
689       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
              %(FullPath)"</Command>
690     </CustomBuild>
691   </ItemGroup>
692   <ItemGroup>
693     <CustomBuild Include="dialog_rem_ide_quant.ui">
694       <FileType>Document</FileType>
695       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
696       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Uic%27ing %(Identity)...</Message>
697       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
698       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
              FullPath)"</Command>
699       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
700       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Uic%27ing %(Identity)...</Message>
701       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
702       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
              %(FullPath)"</Command>
703     </CustomBuild>
704   </ItemGroup>
705   <ItemGroup>
706     <CustomBuild Include="dialog_renomear.ui">
707       <FileType>Document</FileType>
708       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
709       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Uic%27ing %(Identity)...</Message>
```

```
710         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
711         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
712         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
713         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
714         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
715         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
716       </CustomBuild>
717     </ItemGroup>
718     <ItemGroup>
719       <CustomBuild Include="dialog_subst_qual.ui">
720         <FileType>Document</FileType>
721         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
722         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
723         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
724         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
725         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
726         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
727         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
728         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
729       </CustomBuild>
730     </ItemGroup>
731     <ItemGroup>
732       <CustomBuild Include="dialog_subst_quant.ui">
733         <FileType>Document</FileType>
734         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
735         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
736         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
737         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
738         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
739         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
740         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
741         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
742       </CustomBuild>
743     </ItemGroup>
744     <ItemGroup>
745       <CustomBuild Include="dialog_importacao_dados.ui">
746         <FileType>Document</FileType>
747         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
748         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
749         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
750         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
751         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
```

```
            AdditionalInputs>
752       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Uic%27ing %(Identity)...</Message>
753       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
754       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
            %(FullPath)"</Command>
755     </CustomBuild>
756   </ItemGroup>
757   <ItemGroup>
758     <CustomBuild Include="dialog_nova_coluna.ui">
759       <FileType>Document</FileType>
760       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
            AdditionalInputs>
761       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Uic%27ing %(Identity)...</Message>
762       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
763       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
            FullPath)"</Command>
764       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
            AdditionalInputs>
765       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Uic%27ing %(Identity)...</Message>
766       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
767       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
            %(FullPath)"</Command>
768     </CustomBuild>
769   </ItemGroup>
770   <ItemGroup>
771     <CustomBuild Include="widget_intervalo_quant.ui">
772       <FileType>Document</FileType>
773       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
            AdditionalInputs>
774       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Uic%27ing %(Identity)...</Message>
775       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
776       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
            FullPath)"</Command>
777       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
            AdditionalInputs>
778       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Uic%27ing %(Identity)...</Message>
779       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
780       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
            %(FullPath)"</Command>
781     </CustomBuild>
782   </ItemGroup>
783   <ItemGroup>
784     <CustomBuild Include="widget_intervalos_quant.ui">
785       <FileType>Document</FileType>
786       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
            AdditionalInputs>
787       <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Uic%27ing %(Identity)...</Message>
788       <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
789       <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
            FullPath)"</Command>
790       <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
            AdditionalInputs>
791       <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Uic%27ing %(Identity)...</Message>
792       <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
793       <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
```

```
            %(FullPath)"</Command>
794       </CustomBuild>
795     </ItemGroup>
796     <ItemGroup>
797       <CustomBuild Include="widget_categorias_qual.ui">
798         <FileType>Document</FileType>
799         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
800         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Uic%27ing %(Identity)...</Message>
801         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
802         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
              FullPath)"</Command>
803         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
804         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Uic%27ing %(Identity)...</Message>
805         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
806         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
              %(FullPath)"</Command>
807       </CustomBuild>
808     </ItemGroup>
809     <ItemGroup>
810       <CustomBuild Include="dialog_escolher_distribuicao.ui">
811         <FileType>Document</FileType>
812         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
813         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Uic%27ing %(Identity)...</Message>
814         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
815         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
              FullPath)"</Command>
816         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
817         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Uic%27ing %(Identity)...</Message>
818         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
819         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
              %(FullPath)"</Command>
820       </CustomBuild>
821     </ItemGroup>
822     <ItemGroup>
823       <CustomBuild Include="widget_distribuicao.ui">
824         <FileType>Document</FileType>
825         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
826         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Uic%27ing %(Identity)...</Message>
827         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
828         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
              FullPath)"</Command>
829         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
              AdditionalInputs>
830         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Uic%27ing %(Identity)...</Message>
831         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
832         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
              %(FullPath)"</Command>
833       </CustomBuild>
834     </ItemGroup>
835     <ItemGroup>
836       <CustomBuild Include="widget_parametro.ui">
837         <FileType>Document</FileType>
```

```
838         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
839         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
840         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
841         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
842         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
843         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
844         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
845         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
846       </CustomBuild>
847     </ItemGroup>
848     <ItemGroup>
849       <CustomBuild Include="widget_escolher_dp.ui">
850         <FileType>Document</FileType>
851         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
852         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
853         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
854         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
855         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
856         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
857         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
858         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
859       </CustomBuild>
860     </ItemGroup>
861     <ItemGroup>
862       <CustomBuild Include="dialog_rem_ide_outliers.ui">
863         <FileType>Document</FileType>
864         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
865         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
866         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
867         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
868         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
869         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
870         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
871         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
872       </CustomBuild>
873     </ItemGroup>
874     <ItemGroup>
875       <CustomBuild Include="widget_param_diag_caixa.ui">
876         <FileType>Document</FileType>
877         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
878         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
879         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
```

```
880         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
881         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
882         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
883         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
884         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
885       </CustomBuild>
886     </ItemGroup>
887     <ItemGroup>
888       <CustomBuild Include="widget_param_zscore.ui">
889         <FileType>Document</FileType>
890         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
891         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
892         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
893         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
894         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
895         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
896         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
897         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
898       </CustomBuild>
899     </ItemGroup>
900     <ItemGroup>
901       <CustomBuild Include="widget_regressao_linear_simples.ui">
902         <FileType>Document</FileType>
903         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
904         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
905         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
906         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
907         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
908         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
909         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
910         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
911       </CustomBuild>
912     </ItemGroup>
913     <ItemGroup>
914       <ResourceCompile Include="gui_analise_tratamento.rc" />
915     </ItemGroup>
916     <ItemGroup>
917       <CustomBuild Include="dialog_escolher_RLS.ui">
918         <FileType>Document</FileType>
919         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
920         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
921         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
922         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
```

```
923         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
924         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
925         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
926         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
927       </CustomBuild>
928     </ItemGroup>
929     <ItemGroup>
930       <CustomBuild Include="widget_tabela_editora.ui">
931         <FileType>Document</FileType>
932         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
                Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
933         <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
                Uic%27ing %(Identity)...</Message>
934         <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
                GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
935         <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
                (QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "%(
                FullPath)"</Command>
936         <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
                |Win32'">$(QTDIR)\bin\uic.exe;%(AdditionalInputs)</
                AdditionalInputs>
937         <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                Uic%27ing %(Identity)...</Message>
938         <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
                >.\GeneratedFiles\ui_%(Filename).h;%(Outputs)</Outputs>
939         <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
                "$(QTDIR)\bin\uic.exe" -o ".\GeneratedFiles\ui_%(Filename).h" "
                %(FullPath)"</Command>
940       </CustomBuild>
941     </ItemGroup>
942     <Import Project="$(VCTargetsPath)\Microsoft.Cpp.targets" />
943     <ImportGroup Label="ExtensionTargets">
944     </ImportGroup>
945     <ProjectExtensions>
946       <VisualStudio>
947         <UserProperties UicDir=".\GeneratedFiles" MocDir=".\GeneratedFiles\$(
                ConfigurationName)" MocOptions="" RccDir=".\GeneratedFiles"
                lupdateOnBuild="0" lupdateOptions="" lreleaseOptions=""
                Qt5Version_x0020_Win32="msvc2010_opengl" />
948       </VisualStudio>
949     </ProjectExtensions>
950 </Project>
```

Listagem B.2: AnaliseTratamento.vcxproj

```
1  ï»¿<?xml version="1.0" encoding="utf-8"?>
2  <Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/
       msbuild/2003">
3    <ItemGroup>
4      <Filter Include="Source Files">
5        <UniqueIdentifier>{4FC737F1-C7A5-4376-A066-2A32D752A2FF}</
             UniqueIdentifier>
6        <Extensions>cpp;cxx;c;def</Extensions>
7      </Filter>
8      <Filter Include="Header Files">
9        <UniqueIdentifier>{93995380-89BD-4b04-88EB-625FBE52EBFB}</
             UniqueIdentifier>
10       <Extensions>h</Extensions>
11     </Filter>
12     <Filter Include="Form Files">
13       <UniqueIdentifier>{99349809-55BA-4b9d-BF79-8FDBB0286EB3}</
             UniqueIdentifier>
14       <Extensions>ui</Extensions>
15     </Filter>
16     <Filter Include="Generated Files">
17       <UniqueIdentifier>{71ED8ED8-ACB9-4CE9-BBE1-E00B30144E11}</
             UniqueIdentifier>
18       <Extensions>moc;h;cpp</Extensions>
19       <SourceControlFiles>False</SourceControlFiles>
20     </Filter>
21     <Filter Include="Generated Files\Debug">
22       <UniqueIdentifier>{596dfbc0-cc9f-420e-89d2-6ec91456a422}</
             UniqueIdentifier>
```

```
23        <Extensions>cpp;moc</Extensions>
24        <SourceControlFiles>False</SourceControlFiles>
25      </Filter>
26      <Filter Include="Generated Files\Release">
27        <UniqueIdentifier>{47a8f2c6-f429-4d8f-8383-81c953bb1713}</
              UniqueIdentifier>
28        <Extensions>cpp;moc</Extensions>
29        <SourceControlFiles>False</SourceControlFiles>
30      </Filter>
31      <Filter Include="Header Files\Forms">
32        <UniqueIdentifier>{b18a8266-ee97-418f-b5e9-d9e1aba6a405}</
              UniqueIdentifier>
33      </Filter>
34      <Filter Include="Source Files\Forms">
35        <UniqueIdentifier>{4eeebd32-8518-4b62-a29a-4fe4b323af54}</
              UniqueIdentifier>
36      </Filter>
37      <Filter Include="Header Files\thread">
38        <UniqueIdentifier>{1dfde23e-18c9-4958-be62-688684543b8c}</
              UniqueIdentifier>
39      </Filter>
40      <Filter Include="Source Files\thread">
41        <UniqueIdentifier>{b4453ff1-9005-435c-9c22-c49255ff7d50}</
              UniqueIdentifier>
42      </Filter>
43      <Filter Include="Header Files\Graficos">
44        <UniqueIdentifier>{6693ca35-09f2-45fa-8ab6-1140928742b3}</
              UniqueIdentifier>
45      </Filter>
46      <Filter Include="Source Files\Graficos">
47        <UniqueIdentifier>{273b1b3f-c76b-4c2d-80d9-615c36534084}</
              UniqueIdentifier>
48      </Filter>
49      <Filter Include="Header Files\Forms\Resource Files">
50        <UniqueIdentifier>{D9D6E242-F8AF-46E4-B9FD-80ECBC20BA3E}</
              UniqueIdentifier>
51        <Extensions>qrc;*</Extensions>
52        <ParseFiles>false</ParseFiles>
53      </Filter>
54    </ItemGroup>
55    <ItemGroup>
56      <ClCompile Include="main.cpp">
57        <Filter>Source Files</Filter>
58      </ClCompile>
59      <ClCompile Include="GeneratedFiles\Debug\moc_tela_inicial.cpp">
60        <Filter>Generated Files\Debug</Filter>
61      </ClCompile>
62      <ClCompile Include="GeneratedFiles\Release\moc_tela_inicial.cpp">
63        <Filter>Generated Files\Release</Filter>
64      </ClCompile>
65      <ClCompile Include="GeneratedFiles\qrc_tela_inicial.cpp">
66        <Filter>Generated Files</Filter>
67      </ClCompile>
68      <ClCompile Include="tela_inicial.cpp">
69        <Filter>Source Files\Forms</Filter>
70      </ClCompile>
71      <ClCompile Include="populador.cpp">
72        <Filter>Source Files</Filter>
73      </ClCompile>
74      <ClCompile Include="item_tabela_editora.cpp">
75        <Filter>Source Files</Filter>
76      </ClCompile>
77      <ClCompile Include="GeneratedFiles\Debug\
            moc_thread_worker_coletar_estatisticas.cpp">
78        <Filter>Generated Files\Debug</Filter>
79      </ClCompile>
80      <ClCompile Include="GeneratedFiles\Release\
            moc_thread_worker_coletar_estatisticas.cpp">
81        <Filter>Generated Files\Release</Filter>
82      </ClCompile>
83      <ClCompile Include="thread_worker_coletar_estatisticas.cpp">
84        <Filter>Source Files\thread</Filter>
85      </ClCompile>
86      <ClCompile Include="diagrama_scatter.cpp">
87        <Filter>Source Files\Graficos</Filter>
88      </ClCompile>
89      <ClCompile Include="diagrama_caixa.cpp">
90        <Filter>Source Files\Graficos</Filter>
91      </ClCompile>
92      <ClCompile Include="widget_intervalo_quant.cpp">
93        <Filter>Source Files\Forms</Filter>
94      </ClCompile>
```

```
95      <ClCompile Include="widget_intervalos_quant.cpp">
96        <Filter>Source Files\Forms</Filter>
97      </ClCompile>
98      <ClCompile Include="widget_categorias_qual.cpp">
99        <Filter>Source Files\Forms</Filter>
100     </ClCompile>
101     <ClCompile Include="GeneratedFiles\Debug\moc_widget_intervalos_quant.
            cpp">
102       <Filter>Generated Files\Debug</Filter>
103     </ClCompile>
104     <ClCompile Include="GeneratedFiles\Release\moc_widget_intervalos_quant.
            cpp">
105       <Filter>Generated Files\Release</Filter>
106     </ClCompile>
107     <ClCompile Include="GeneratedFiles\Debug\moc_widget_intervalo_quant.cpp
            ">
108       <Filter>Generated Files\Debug</Filter>
109     </ClCompile>
110     <ClCompile Include="GeneratedFiles\Release\moc_widget_intervalo_quant.
            cpp">
111       <Filter>Generated Files\Release</Filter>
112     </ClCompile>
113     <ClCompile Include="GeneratedFiles\Debug\moc_widget_categorias_qual.cpp
            ">
114       <Filter>Generated Files\Debug</Filter>
115     </ClCompile>
116     <ClCompile Include="GeneratedFiles\Release\moc_widget_categorias_qual.
            cpp">
117       <Filter>Generated Files\Release</Filter>
118     </ClCompile>
119     <ClCompile Include="GeneratedFiles\Debug\moc_widget_distribuicao.cpp">
120       <Filter>Generated Files\Debug</Filter>
121     </ClCompile>
122     <ClCompile Include="GeneratedFiles\Release\moc_widget_distribuicao.cpp"
            >
123       <Filter>Generated Files\Release</Filter>
124     </ClCompile>
125     <ClCompile Include="GeneratedFiles\Debug\moc_widget_parametro.cpp">
126       <Filter>Generated Files\Debug</Filter>
127     </ClCompile>
128     <ClCompile Include="GeneratedFiles\Release\moc_widget_parametro.cpp">
129       <Filter>Generated Files\Release</Filter>
130     </ClCompile>
131     <ClCompile Include="widget_parametro.cpp">
132       <Filter>Source Files\Forms</Filter>
133     </ClCompile>
134     <ClCompile Include="widget_distribuicao.cpp">
135       <Filter>Source Files\Forms</Filter>
136     </ClCompile>
137     <ClCompile Include="GeneratedFiles\Debug\moc_widget_escolher_dp.cpp">
138       <Filter>Generated Files\Debug</Filter>
139     </ClCompile>
140     <ClCompile Include="GeneratedFiles\Release\moc_widget_escolher_dp.cpp">
141       <Filter>Generated Files\Release</Filter>
142     </ClCompile>
143     <ClCompile Include="widget_escolher_dp.cpp">
144       <Filter>Source Files\Forms</Filter>
145     </ClCompile>
146     <ClCompile Include="GeneratedFiles\Debug\moc_widget_param_diag_caixa.
            cpp">
147       <Filter>Generated Files\Debug</Filter>
148     </ClCompile>
149     <ClCompile Include="GeneratedFiles\Release\moc_widget_param_diag_caixa.
            cpp">
150       <Filter>Generated Files\Release</Filter>
151     </ClCompile>
152     <ClCompile Include="widget_param_diag_caixa.cpp">
153       <Filter>Source Files\Forms</Filter>
154     </ClCompile>
155     <ClCompile Include="GeneratedFiles\Debug\moc_widget_param_zscore.cpp">
156       <Filter>Generated Files\Debug</Filter>
157     </ClCompile>
158     <ClCompile Include="GeneratedFiles\Release\moc_widget_param_zscore.cpp"
            >
159       <Filter>Generated Files\Release</Filter>
160     </ClCompile>
161     <ClCompile Include="widget_param_zscore.cpp">
162       <Filter>Source Files\Forms</Filter>
163     </ClCompile>
164     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_edicao_dados.cpp">
165       <Filter>Generated Files\Debug</Filter>
166     </ClCompile>
```

```
167     <ClCompile Include="GeneratedFiles\Release\moc_dialog_edicao_dados.cpp"
            >
168       <Filter>Generated Files\Release</Filter>
169     </ClCompile>
170     <ClCompile Include="dialog_edicao_dados.cpp">
171       <Filter>Source Files\Forms</Filter>
172     </ClCompile>
173     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_rem_ide_quali.cpp">
174       <Filter>Generated Files\Debug</Filter>
175     </ClCompile>
176     <ClCompile Include="GeneratedFiles\Release\moc_dialog_rem_ide_quali.cpp
            ">
177       <Filter>Generated Files\Release</Filter>
178     </ClCompile>
179     <ClCompile Include="dialog_rem_ide_quali.cpp">
180       <Filter>Source Files\Forms</Filter>
181     </ClCompile>
182     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_rem_ide_quant.cpp">
183       <Filter>Generated Files\Debug</Filter>
184     </ClCompile>
185     <ClCompile Include="GeneratedFiles\Release\moc_dialog_rem_ide_quant.cpp
            ">
186       <Filter>Generated Files\Release</Filter>
187     </ClCompile>
188     <ClCompile Include="dialog_rem_ide_quant.cpp">
189       <Filter>Source Files\Forms</Filter>
190     </ClCompile>
191     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_renomear.cpp">
192       <Filter>Generated Files\Debug</Filter>
193     </ClCompile>
194     <ClCompile Include="GeneratedFiles\Release\moc_dialog_renomear.cpp">
195       <Filter>Generated Files\Release</Filter>
196     </ClCompile>
197     <ClCompile Include="dialog_renomear.cpp">
198       <Filter>Source Files\Forms</Filter>
199     </ClCompile>
200     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_subst_qual.cpp">
201       <Filter>Generated Files\Debug</Filter>
202     </ClCompile>
203     <ClCompile Include="GeneratedFiles\Release\moc_dialog_subst_qual.cpp">
204       <Filter>Generated Files\Release</Filter>
205     </ClCompile>
206     <ClCompile Include="dialog_subst_qual.cpp">
207       <Filter>Source Files\Forms</Filter>
208     </ClCompile>
209     <ClCompile Include="dialog_subst_quant.cpp">
210       <Filter>Source Files\Forms</Filter>
211     </ClCompile>
212     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_subst_quant.cpp">
213       <Filter>Generated Files\Debug</Filter>
214     </ClCompile>
215     <ClCompile Include="GeneratedFiles\Release\moc_dialog_subst_quant.cpp">
216       <Filter>Generated Files\Release</Filter>
217     </ClCompile>
218     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_nova_coluna.cpp">
219       <Filter>Generated Files\Debug</Filter>
220     </ClCompile>
221     <ClCompile Include="GeneratedFiles\Release\moc_dialog_nova_coluna.cpp">
222       <Filter>Generated Files\Release</Filter>
223     </ClCompile>
224     <ClCompile Include="dialog_nova_coluna.cpp">
225       <Filter>Source Files\Forms</Filter>
226     </ClCompile>
227     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_importacao_dados.
            cpp">
228       <Filter>Generated Files\Debug</Filter>
229     </ClCompile>
230     <ClCompile Include="GeneratedFiles\Release\moc_dialog_importacao_dados.
            cpp">
231       <Filter>Generated Files\Release</Filter>
232     </ClCompile>
233     <ClCompile Include="dialog_importacao_dados.cpp">
234       <Filter>Source Files\Forms</Filter>
235     </ClCompile>
236     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_visualizar_graficos
            .cpp">
237       <Filter>Generated Files\Debug</Filter>
238     </ClCompile>
239     <ClCompile Include="GeneratedFiles\Release\
            moc_dialog_visualizar_graficos.cpp">
240       <Filter>Generated Files\Release</Filter>
241     </ClCompile>
```

```
242     <ClCompile Include="dialog_visualizar_graficos.cpp">
243       <Filter>Source Files\Forms</Filter>
244     </ClCompile>
245     <ClCompile Include="dialog_escolher_distribuicao.cpp">
246       <Filter>Source Files\Forms</Filter>
247     </ClCompile>
248     <ClCompile Include="GeneratedFiles\Debug\
            moc_dialog_escolher_distribuicao.cpp">
249       <Filter>Generated Files\Debug</Filter>
250     </ClCompile>
251     <ClCompile Include="GeneratedFiles\Release\
            moc_dialog_escolher_distribuicao.cpp">
252       <Filter>Generated Files\Release</Filter>
253     </ClCompile>
254     <ClCompile Include="dialog_rem_ide_outliers.cpp">
255       <Filter>Source Files\Forms</Filter>
256     </ClCompile>
257     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_rem_ide_outliers.
            cpp">
258       <Filter>Generated Files\Debug</Filter>
259     </ClCompile>
260     <ClCompile Include="GeneratedFiles\Release\moc_dialog_rem_ide_outliers.
            cpp">
261       <Filter>Generated Files\Release</Filter>
262     </ClCompile>
263     <ClCompile Include="widget_analise_exploracao_dados.cpp">
264       <Filter>Source Files\Forms</Filter>
265     </ClCompile>
266     <ClCompile Include="GeneratedFiles\Debug\
            moc_widget_analise_exploracao_dados.cpp">
267       <Filter>Generated Files\Debug</Filter>
268     </ClCompile>
269     <ClCompile Include="GeneratedFiles\Release\
            moc_widget_analise_exploracao_dados.cpp">
270       <Filter>Generated Files\Release</Filter>
271     </ClCompile>
272     <ClCompile Include="histograma_quant_continuo.cpp">
273       <Filter>Source Files\Graficos</Filter>
274     </ClCompile>
275     <ClCompile Include="GeneratedFiles\Debug\moc_histograma_quant_continuo.
            cpp">
276       <Filter>Generated Files\Debug</Filter>
277     </ClCompile>
278     <ClCompile Include="GeneratedFiles\Release\
            moc_histograma_quant_continuo.cpp">
279       <Filter>Generated Files\Release</Filter>
280     </ClCompile>
281     <ClCompile Include="GeneratedFiles\Debug\
            moc_widget_regressao_linear_simples.cpp">
282       <Filter>Generated Files\Debug</Filter>
283     </ClCompile>
284     <ClCompile Include="GeneratedFiles\Release\
            moc_widget_regressao_linear_simples.cpp">
285       <Filter>Generated Files\Release</Filter>
286     </ClCompile>
287     <ClCompile Include="widget_regressao_linear_simples.cpp">
288       <Filter>Source Files\Forms</Filter>
289     </ClCompile>
290     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_escolher_RLS.cpp">
291       <Filter>Generated Files\Debug</Filter>
292     </ClCompile>
293     <ClCompile Include="GeneratedFiles\Release\moc_dialog_escolher_RLS.cpp"
            >
294       <Filter>Generated Files\Release</Filter>
295     </ClCompile>
296     <ClCompile Include="dialog_escolher_RLS.cpp">
297       <Filter>Source Files\Forms</Filter>
298     </ClCompile>
299     <ClCompile Include="GeneratedFiles\Debug\moc_widget_tabela_editora.cpp"
            >
300       <Filter>Generated Files\Debug</Filter>
301     </ClCompile>
302     <ClCompile Include="GeneratedFiles\Release\moc_widget_tabela_editora.
            cpp">
303       <Filter>Generated Files\Release</Filter>
304     </ClCompile>
305     <ClCompile Include="widget_tabela_editora.cpp">
306       <Filter>Source Files\Forms</Filter>
307     </ClCompile>
308     <ClCompile Include="dialog_importacao_dados_pg_00_arquivo.cpp">
309       <Filter>Source Files\Forms</Filter>
310     </ClCompile>
```

```
311     <ClCompile Include="dialog_importacao_dados_pg_01_selecionar_atribs.cpp
            ">
312       <Filter>Source Files\Forms</Filter>
313     </ClCompile>
314     <ClCompile Include="dialog_importacao_dados_pg_02_tipificar.cpp">
315       <Filter>Source Files\Forms</Filter>
316     </ClCompile>
317     <ClCompile Include="dialog_importacao_dados_pg_03_def_interv_cat.cpp">
318       <Filter>Source Files\Forms</Filter>
319     </ClCompile>
320     <ClCompile Include="
            dialog_importacao_dados_pg_04_tratar_fora_interv_cat.cpp">
321       <Filter>Source Files\Forms</Filter>
322     </ClCompile>
323     <ClCompile Include="dialog_importacao_dados_pg_05_tec_outliers.cpp">
324       <Filter>Source Files\Forms</Filter>
325     </ClCompile>
326     <ClCompile Include="dialog_importacao_dados_pg_06_tratar_outliers.cpp">
327       <Filter>Source Files\Forms</Filter>
328     </ClCompile>
329     <ClCompile Include="
            dialog_importacao_dados_pg_07_tratar_missings_invalidos.cpp">
330       <Filter>Source Files\Forms</Filter>
331     </ClCompile>
332     <ClCompile Include="GeneratedFiles\Debug\moc_grafico_multibarras.cpp">
333       <Filter>Generated Files\Debug</Filter>
334     </ClCompile>
335     <ClCompile Include="GeneratedFiles\Release\moc_grafico_multibarras.cpp"
            >
336       <Filter>Generated Files\Release</Filter>
337     </ClCompile>
338     <ClCompile Include="grafico_multibarras.cpp">
339       <Filter>Source Files\Graficos</Filter>
340     </ClCompile>
341     <ClCompile Include="GeneratedFiles\Debug\
            moc_grafico_barras_qual_quant_discreto.cpp">
342       <Filter>Generated Files\Debug</Filter>
343     </ClCompile>
344     <ClCompile Include="GeneratedFiles\Release\
            moc_grafico_barras_qual_quant_discreto.cpp">
345       <Filter>Generated Files\Release</Filter>
346     </ClCompile>
347     <ClCompile Include="grafico_barras_qual_quant_discreto.cpp">
348       <Filter>Source Files\Graficos</Filter>
349     </ClCompile>
350   </ItemGroup>
351   <ItemGroup>
352     <CustomBuild Include="tela_inicial.ui">
353       <Filter>Form Files</Filter>
354     </CustomBuild>
355     <CustomBuild Include="tela_inicial.qrc">
356       <Filter>Header Files\Forms\Resource Files</Filter>
357     </CustomBuild>
358     <CustomBuild Include="tela_inicial.h">
359       <Filter>Header Files\Forms</Filter>
360     </CustomBuild>
361     <CustomBuild Include="thread_worker_coletar_estatisticas.h">
362       <Filter>Header Files\thread</Filter>
363     </CustomBuild>
364     <CustomBuild Include="widget_intervalo_quant.ui">
365       <Filter>Form Files</Filter>
366     </CustomBuild>
367     <CustomBuild Include="widget_intervalos_quant.ui">
368       <Filter>Form Files</Filter>
369     </CustomBuild>
370     <CustomBuild Include="widget_categorias_qual.ui">
371       <Filter>Form Files</Filter>
372     </CustomBuild>
373     <CustomBuild Include="widget_intervalos_quant.h">
374       <Filter>Header Files\Forms</Filter>
375     </CustomBuild>
376     <CustomBuild Include="widget_intervalo_quant.h">
377       <Filter>Header Files\Forms</Filter>
378     </CustomBuild>
379     <CustomBuild Include="widget_categorias_qual.h">
380       <Filter>Header Files\Forms</Filter>
381     </CustomBuild>
382     <CustomBuild Include="widget_distribuicao.ui">
383       <Filter>Form Files</Filter>
384     </CustomBuild>
385     <CustomBuild Include="widget_parametro.ui">
386       <Filter>Form Files</Filter>
```

```
387       </CustomBuild>
388       <CustomBuild Include="widget_parametro.h">
389         <Filter>Header Files\Forms</Filter>
390       </CustomBuild>
391       <CustomBuild Include="widget_distribuicao.h">
392         <Filter>Header Files\Forms</Filter>
393       </CustomBuild>
394       <CustomBuild Include="widget_escolher_dp.ui">
395         <Filter>Form Files</Filter>
396       </CustomBuild>
397       <CustomBuild Include="widget_escolher_dp.h">
398         <Filter>Header Files\Forms</Filter>
399       </CustomBuild>
400       <CustomBuild Include="widget_param_diag_caixa.ui">
401         <Filter>Form Files</Filter>
402       </CustomBuild>
403       <CustomBuild Include="widget_param_diag_caixa.h">
404         <Filter>Header Files\Forms</Filter>
405       </CustomBuild>
406       <CustomBuild Include="widget_param_zscore.ui">
407         <Filter>Form Files</Filter>
408       </CustomBuild>
409       <CustomBuild Include="widget_param_zscore.h">
410         <Filter>Header Files\Forms</Filter>
411       </CustomBuild>
412       <CustomBuild Include="dialog_edicao_dados.ui">
413         <Filter>Form Files</Filter>
414       </CustomBuild>
415       <CustomBuild Include="dialog_edicao_dados.h">
416         <Filter>Header Files\Forms</Filter>
417       </CustomBuild>
418       <CustomBuild Include="dialog_rem_ide_quali.ui">
419         <Filter>Form Files</Filter>
420       </CustomBuild>
421       <CustomBuild Include="dialog_rem_ide_quali.h">
422         <Filter>Header Files\Forms</Filter>
423       </CustomBuild>
424       <CustomBuild Include="dialog_rem_ide_quant.ui">
425         <Filter>Form Files</Filter>
426       </CustomBuild>
427       <CustomBuild Include="dialog_rem_ide_quant.h">
428         <Filter>Header Files\Forms</Filter>
429       </CustomBuild>
430       <CustomBuild Include="dialog_renomear.ui">
431         <Filter>Form Files</Filter>
432       </CustomBuild>
433       <CustomBuild Include="dialog_renomear.h">
434         <Filter>Header Files\Forms</Filter>
435       </CustomBuild>
436       <CustomBuild Include="dialog_subst_qual.ui">
437         <Filter>Form Files</Filter>
438       </CustomBuild>
439       <CustomBuild Include="dialog_subst_qual.h">
440         <Filter>Header Files\Forms</Filter>
441       </CustomBuild>
442       <CustomBuild Include="dialog_subst_quant.ui">
443         <Filter>Form Files</Filter>
444       </CustomBuild>
445       <CustomBuild Include="dialog_subst_quant.h">
446         <Filter>Header Files\Forms</Filter>
447       </CustomBuild>
448       <CustomBuild Include="dialog_nova_coluna.ui">
449         <Filter>Form Files</Filter>
450       </CustomBuild>
451       <CustomBuild Include="dialog_nova_coluna.h">
452         <Filter>Header Files\Forms</Filter>
453       </CustomBuild>
454       <CustomBuild Include="dialog_importacao_dados.ui">
455         <Filter>Form Files</Filter>
456       </CustomBuild>
457       <CustomBuild Include="dialog_importacao_dados.h">
458         <Filter>Header Files\Forms</Filter>
459       </CustomBuild>
460       <CustomBuild Include="dialog_visualizar_graficos.ui">
461         <Filter>Form Files</Filter>
462       </CustomBuild>
463       <CustomBuild Include="dialog_visualizar_graficos.h">
464         <Filter>Header Files\Forms</Filter>
465       </CustomBuild>
466       <CustomBuild Include="dialog_escolher_distribuicao.ui">
467         <Filter>Form Files</Filter>
468       </CustomBuild>
```

```
469     <CustomBuild Include="dialog_escolher_distribuicao.h">
470       <Filter>Header Files\Forms</Filter>
471     </CustomBuild>
472     <CustomBuild Include="dialog_rem_ide_outliers.ui">
473       <Filter>Form Files</Filter>
474     </CustomBuild>
475     <CustomBuild Include="dialog_rem_ide_outliers.h">
476       <Filter>Header Files\Forms</Filter>
477     </CustomBuild>
478     <CustomBuild Include="widget_analise_exploracao_dados.ui">
479       <Filter>Form Files</Filter>
480     </CustomBuild>
481     <CustomBuild Include="widget_analise_exploracao_dados.h">
482       <Filter>Header Files\Forms</Filter>
483     </CustomBuild>
484     <CustomBuild Include="histograma_quant_continuo.h">
485       <Filter>Header Files\Graficos</Filter>
486     </CustomBuild>
487     <CustomBuild Include="widget_regressao_linear_simples.ui">
488       <Filter>Form Files</Filter>
489     </CustomBuild>
490     <CustomBuild Include="widget_regressao_linear_simples.h">
491       <Filter>Header Files\Forms</Filter>
492     </CustomBuild>
493     <CustomBuild Include="dialog_escolher_RLS.ui">
494       <Filter>Form Files</Filter>
495     </CustomBuild>
496     <CustomBuild Include="dialog_escolher_RLS.h">
497       <Filter>Header Files\Forms</Filter>
498     </CustomBuild>
499     <CustomBuild Include="widget_tabela_editora.ui">
500       <Filter>Form Files</Filter>
501     </CustomBuild>
502     <CustomBuild Include="widget_tabela_editora.h">
503       <Filter>Header Files\Forms</Filter>
504     </CustomBuild>
505     <CustomBuild Include="grafico_multibarras.h">
506       <Filter>Header Files\Graficos</Filter>
507     </CustomBuild>
508     <CustomBuild Include="grafico_barras_qual_quant_discreto.h">
509       <Filter>Header Files\Graficos</Filter>
510     </CustomBuild>
511   </ItemGroup>
512   <ItemGroup>
513     <ClInclude Include="GeneratedFiles\ui_tela_inicial.h">
514       <Filter>Generated Files</Filter>
515     </ClInclude>
516     <ClInclude Include="item_tabela_editora.h">
517       <Filter>Header Files</Filter>
518     </ClInclude>
519     <ClInclude Include="escala_nomes.h">
520       <Filter>Header Files</Filter>
521     </ClInclude>
522     <ClInclude Include="diagrama_scatter.h">
523       <Filter>Header Files\Graficos</Filter>
524     </ClInclude>
525     <ClInclude Include="diagrama_caixa.h">
526       <Filter>Header Files\Graficos</Filter>
527     </ClInclude>
528     <ClInclude Include="GeneratedFiles\ui_widget_intervalo_quant.h">
529       <Filter>Generated Files</Filter>
530     </ClInclude>
531     <ClInclude Include="GeneratedFiles\ui_widget_intervalos_quant.h">
532       <Filter>Generated Files</Filter>
533     </ClInclude>
534     <ClInclude Include="GeneratedFiles\ui_widget_categorias_qual.h">
535       <Filter>Generated Files</Filter>
536     </ClInclude>
537     <ClInclude Include="GeneratedFiles\ui_widget_distribuicao.h">
538       <Filter>Generated Files</Filter>
539     </ClInclude>
540     <ClInclude Include="GeneratedFiles\ui_widget_parametro.h">
541       <Filter>Generated Files</Filter>
542     </ClInclude>
543     <ClInclude Include="GeneratedFiles\ui_widget_escolher_dp.h">
544       <Filter>Generated Files</Filter>
545     </ClInclude>
546     <ClInclude Include="GeneratedFiles\ui_widget_param_diag_caixa.h">
547       <Filter>Generated Files</Filter>
548     </ClInclude>
549     <ClInclude Include="GeneratedFiles\ui_widget_param_zscore.h">
550       <Filter>Generated Files</Filter>
```

```
551     </ClInclude>
552     <ClInclude Include="populador.h">
553       <Filter>Header Files</Filter>
554     </ClInclude>
555     <ClInclude Include="GeneratedFiles\ui_dialog_edicao_dados.h">
556       <Filter>Generated Files</Filter>
557     </ClInclude>
558     <ClInclude Include="GeneratedFiles\ui_dialog_rem_ide_quali.h">
559       <Filter>Generated Files</Filter>
560     </ClInclude>
561     <ClInclude Include="GeneratedFiles\ui_dialog_rem_ide_quant.h">
562       <Filter>Generated Files</Filter>
563     </ClInclude>
564     <ClInclude Include="GeneratedFiles\ui_dialog_renomear.h">
565       <Filter>Generated Files</Filter>
566     </ClInclude>
567     <ClInclude Include="GeneratedFiles\ui_dialog_subst_qual.h">
568       <Filter>Generated Files</Filter>
569     </ClInclude>
570     <ClInclude Include="GeneratedFiles\ui_dialog_subst_quant.h">
571       <Filter>Generated Files</Filter>
572     </ClInclude>
573     <ClInclude Include="GeneratedFiles\ui_dialog_nova_coluna.h">
574       <Filter>Generated Files</Filter>
575     </ClInclude>
576     <ClInclude Include="GeneratedFiles\ui_dialog_importacao_dados.h">
577       <Filter>Generated Files</Filter>
578     </ClInclude>
579     <ClInclude Include="GeneratedFiles\ui_dialog_visualizar_graficos.h">
580       <Filter>Generated Files</Filter>
581     </ClInclude>
582     <ClInclude Include="GeneratedFiles\ui_dialog_escolher_distribuicao.h">
583       <Filter>Generated Files</Filter>
584     </ClInclude>
585     <ClInclude Include="GeneratedFiles\ui_dialog_rem_ide_outliers.h">
586       <Filter>Generated Files</Filter>
587     </ClInclude>
588     <ClInclude Include="GeneratedFiles\ui_widget_analise_exploracao_dados.h
          ">
589       <Filter>Generated Files</Filter>
590     </ClInclude>
591     <ClInclude Include="GeneratedFiles\ui_widget_regressao_linear_simples.h
          ">
592       <Filter>Generated Files</Filter>
593     </ClInclude>
594     <ClInclude Include="resource.h">
595       <Filter>Header Files</Filter>
596     </ClInclude>
597     <ClInclude Include="GeneratedFiles\ui_dialog_escolher_RLS.h">
598       <Filter>Generated Files</Filter>
599     </ClInclude>
600     <ClInclude Include="GeneratedFiles\ui_widget_tabela_editora.h">
601       <Filter>Generated Files</Filter>
602     </ClInclude>
603   </ItemGroup>
604   <ItemGroup>
605     <ResourceCompile Include="gui_analise_tratamento.rc" />
606   </ItemGroup>
607 </Project>
```

Listagem B.3: AnaliseTratamento.vcxproj.filters

```
1  // Microsoft Visual C++ generated resource script.
2  //
3  #include "resource.h"
4
5  #define APSTUDIO_READONLY_SYMBOLS
6  //
       //////////////////////////////////////////////////////////////////////////////
7  //
8  // Generated from the TEXTINCLUDE 2 resource.
9  //
10 #include "afxres.h"
11
12 //
       //////////////////////////////////////////////////////////////////////////////
13 #undef APSTUDIO_READONLY_SYMBOLS
```

```
14
15 //
       /////////////////////////////////////////////////////////////////////////////
16 // Portuguese (Brazil) resources
17
18 #if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_PTB)
19 LANGUAGE LANG_PORTUGUESE, SUBLANG_PORTUGUESE_BRAZILIAN
20
21 #ifdef APSTUDIO_INVOKED
22 //
       /////////////////////////////////////////////////////////////////////////////
23 //
24 // TEXTINCLUDE
25 //
26
27 1 TEXTINCLUDE
28 BEGIN
29     "resource.h\0"
30 END
31
32 2 TEXTINCLUDE
33 BEGIN
34     "#include ""afxres.h""\r\n"
35     "\0"
36 END
37
38 3 TEXTINCLUDE
39 BEGIN
40     "\r\n"
41     "\0"
42 END
43
44 #endif    // APSTUDIO_INVOKED
45
46 #endif    // Portuguese (Brazil) resources
47 //
       /////////////////////////////////////////////////////////////////////////////
48
49
50
51 #ifndef APSTUDIO_INVOKED
52 //
       /////////////////////////////////////////////////////////////////////////////
53 //
54 // Generated from the TEXTINCLUDE 3 resource.
55 //
56
57
58 //
       /////////////////////////////////////////////////////////////////////////////
59 #endif    // not APSTUDIO_INVOKED
```

Listagem B.4: gui_analise_tratamento.rc

```
1  //{{NO_DEPENDENCIES}}
2  // Microsoft Visual C++ generated include file.
3  // Used by gui_analise_tratamento.rc
4  //
5
6  // Next default values for new objects
7  //
8  #ifdef APSTUDIO_INVOKED
9  #ifndef APSTUDIO_READONLY_SYMBOLS
10 #define _APS_NEXT_RESOURCE_VALUE        103
11 #define _APS_NEXT_COMMAND_VALUE         40001
12 #define _APS_NEXT_CONTROL_VALUE         1001
13 #define _APS_NEXT_SYMED_VALUE           101
14 #endif
15 #endif
```

Listagem B.5: resource.h

```
#include "tela_inicial.h"

#include <QtWidgets/QApplication>
#include <fstream>

//#include <vld.h>

/*#define _CRTDBG_MAP_ALLOC
#include<iostream>
#include <crtdbg.h>
#define DEBUG_NEW new(_NORMAL_BLOCK, __FILE__, __LINE__)
#define new DEBUG_NEW*/

void crashMessageOutput(QtMsgType type, const QMessageLogContext&, const
     QString &str)
{
    std::ofstream out;

    //in this function, you can write the message to any stream!
    switch (type) {
    case QtDebugMsg:
        try
        {
            out.open("gui_analise_tratamento.log", std::ios::app);
            if (out.is_open())
            {
                out << "Debug: " << str.toStdString() << "\n";
                out.close();
            }
        }
        catch(std::ifstream::failure e){}

        break;
    case QtWarningMsg:
        break;
    case QtCriticalMsg:
        break;
    case QtFatalMsg:
        break;
    }
}

int main(int argc, char *argv[])
{
    remove("gui_analise_tratamento.log");
    qInstallMessageHandler(crashMessageOutput);

    QLocale::setDefault(QLocale(QLocale::Portuguese, QLocale::Brazil));

    QApplication a(argc, argv);
    TelaInicial w = TelaInicial();
    w.show();
    int result = a.exec();

    //_CrtDumpMemoryLeaks();

    return result;
}
```

Listagem B.6: main.cpp

```
#ifndef ESCALANOME_H
#define ESCALANOME_H

#include <qwt_scale_draw.h>
#include <vector>
#include <string>
#include <string>

#include "utilidades.h"

using namespace std;

class EscalaNomes : public QwtScaleDraw
{
private:
    vector<string> _atributos;
public:
    EscalaNomes() : QwtScaleDraw()
```

```
19      {
20      }
21
22      EscalaNomes(const vector<string> &atributos) : QwtScaleDraw()
23      {
24          this->_atributos = atributos;
25      }
26
27      void setAtributos(const vector<string> &atributos)
28      {
29          this->_atributos = atributos;
30      }
31
32      virtual ~EscalaNomes() {}
33      virtual QwtText label(double b) const
34      {
35          if (_atributos.empty())
36          {
37              return  QwtText(Utilidades::dblToStdStr(b, 10).c_str());
38          }
39          else
40          {
41              int valueInt = b;
42
43              if (valueInt >= _atributos.size() || valueInt < 0)
44              {
45                  return QwtText("");
46              }
47
48              return QwtText(QString::fromLatin1(_atributos[valueInt].c_str()
                        ));
49          }
50      }
51  };
52
53  #endif
```

Listagem B.7: escala_nomes.h

```
1   #include "grafico_multibarras.h"
2
3   #include <qwt_plot_renderer.h>
4   #include <qwt_plot_canvas.h>
5   #include <qwt_plot_multi_barchart.h>
6   #include <qwt_column_symbol.h>
7   #include <qwt_plot_layout.h>
8   #include <qwt_legend.h>
9   #include <qwt_scale_draw.h>
10  #include <qwt_plot_grid.h>
11  #include <qwt_plot_magnifier.h>
12  #include <qwt_plot_panner.h>
13
14  #include "utilidades.h"
15
16  GraficoMultiBarras::GraficoMultiBarras(QWidget *parent):
17  QwtPlot(parent)
18  {
19      setAutoFillBackground(true);
20      QPalette p = palette();
21      p.setColor(QPalette::Window, Qt::transparent);
22      setPalette(p);
23
24      setAutoReplot(false);
25
26      canvas()->setPalette(Qt::white);
27
28      d_barChartItem = new QwtPlotMultiBarChart();
29      d_barChartItem->setLayoutPolicy(QwtPlotMultiBarChart::AutoAdjustSamples
            );
30      d_barChartItem->setSpacing(20);
31      d_barChartItem->setMargin(3);
32
33      d_barChartItem->attach(this);
34
35      setEstiloPilha();
36
37      insertLegend(new QwtLegend());
38      _tituloLegenda = new QwtPlotTextLabel();
39      _tituloLegenda->attach(this);
```

```
40
41      QwtPlotGrid *grid = new QwtPlotGrid();
42      grid->attach(this);
43      grid->setPen(QPen(QColor(200, 200, 200)));
44
45      // panning with the left mouse button
46      (void )new QwtPlotPanner( canvas() );
47      // zoom in/out with the wheel
48      QwtPlotMagnifier *magnifier = new QwtPlotMagnifier(canvas());
49      magnifier->setMouseButton(Qt::NoButton);
50      magnifier->setWheelFactor(-1.1);
51  }
52
53  GraficoMultiBarras::~GraficoMultiBarras()
54  {
55      const void * address = static_cast<const void*>(&(*this));
56      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
57
58      delete _tituloLegenda;
59  }
60
61  void GraficoMultiBarras::populate(const vector<vector<int>> &frequencias,
        const vector<string> &titulos)
62  {
63      const size_t numSamples = frequencias.size();
64      const size_t numBars = frequencias[0].size();
65
66      QList<QwtText> titles = QList<QwtText>();
67      for (vector<string>::const_iterator citTitulo = titulos.cbegin();
68              citTitulo != titulos.cend();
69              citTitulo++)
70      {
71          titles += QString::fromLatin1(citTitulo->c_str());
72      }
73
74      d_barChartItem->setBarTitles(titles);
75      d_barChartItem->setLegendIconSize(QSize(10, 14));
76
77      for (int i = 0; i < numBars; i++)
78      {
79          QwtColumnSymbol *symbol = new QwtColumnSymbol(QwtColumnSymbol::Box)
                ;
80          symbol->setLineWidth(2);
81          symbol->setFrameStyle(QwtColumnSymbol::Raised);
82          symbol->setPalette(QPalette(QColor(Utilidades::coresDistintas[i %
                56].c_str())));
83          d_barChartItem->setSymbol(i, symbol);
84      }
85
86      QVector< QVector<double> > series;
87      for (int i = 0; i < numSamples; i++)
88      {
89          QVector<double> values;
90          for (int j = 0; j < numBars; j++)
91          {
92              values += (double)frequencias[i][j];
93          }
94
95          series += values;
96      }
97
98      d_barChartItem->setSamples(series);
99  }
100
101 void GraficoMultiBarras::ajustarLabelEixo(int axisId, Qt::Alignment
        alinhamento, int rotacao)
102 {
103     setAxisLabelAlignment(axisId, alinhamento);
104     setAxisLabelRotation(axisId, rotacao);
105 }
106
107 void GraficoMultiBarras::setTituloLegenda(QString titulo)
108 {
109     if (_tituloLegenda)
110     {
111         QwtText title(titulo);
112         title.setRenderFlags( Qt::AlignRight | Qt::AlignTop );
113         title.setFont(axisTitle(QwtPlot::xBottom).font());
114         _tituloLegenda->setText(title);
115     }
116 }
117
```

```
118 void GraficoMultiBarras::setEstiloBarras()
119 {
120     d_barChartItem->setStyle(QwtPlotMultiBarChart::Grouped);
121 }
122
123 void GraficoMultiBarras::setEstiloPilha()
124 {
125     d_barChartItem->setStyle(QwtPlotMultiBarChart::Stacked);
126 }
```

Listagem B.8: grafico_multibarras.cpp

```
 1 #ifndef GRAFICO_MULTI_BARRAS
 2 #define GRAFICO_MULTI_BARRAS
 3
 4 #include <qwt_plot.h>
 5 #include <qwt_plot_textlabel.h>
 6 #include <vector>
 7
 8 using namespace std;
 9
10 class QwtPlotMultiBarChart;
11 class GraficoMultiBarras: public QwtPlot
12 {
13     Q_OBJECT
14
15 public:
16     GraficoMultiBarras(QWidget *parent = NULL);
17     ~GraficoMultiBarras();
18     void populate(const vector<vector<int>> &frequencias, const vector<
           string> &titulos);
19     void ajustarLabelEixo(int axisId, Qt::Alignment alinhamento, int
           rotacao);
20     void setTituloLegenda(QString titulo);
21     void setEstiloBarras();
22     void setEstiloPilha();
23
24 private:
25     QwtPlotMultiBarChart *d_barChartItem;
26     QwtPlotTextLabel *_tituloLegenda;
27 };
28
29 #endif
```

Listagem B.9: grafico_multibarras.h

```
 1 #include "grafico_barras_qual_quant_discreto.h"
 2
 3 #include <qwt_plot_grid.h>
 4
 5 GraficoBarrasQualQuantDiscreto::GraficoBarrasQualQuantDiscreto(
 6     const vector<string> &categorias,
 7     const vector<int> &frequencias,
 8     const int count,
 9     QWidget *parent):
10 QwtPlot(parent)
11 {
12     setAutoReplot(false);
13
14     setUpdatesEnabled(false);
15     setAxisAutoScale(false);
16
17     setAutoFillBackground(true);
18
19     d_barChartItem = new QwtPlotBarChart();
20     d_barChartItem->setLayoutPolicy(QwtPlotBarChart::AutoAdjustSamples);
21     d_barChartItem->setSpacing(3);
22     d_barChartItem->setMargin(3);
23
24     const size_t max = categorias.size();
25     setAxisScale(QwtPlot::xBottom, -1, max, 1);
26     updateAxes();
27
28     setAxisScaleDraw(QwtPlot::xBottom, new EscalaNomes(categorias));
29     updateAxes();
30
```

```
31      QFont font;
32      font.setPointSize(8);
33
34      setAxisFont(QwtPlot::xBottom, font);
35      setAxisLabelAlignment(QwtPlot::xBottom, Qt::AlignRight);
36      setAxisLabelRotation(QwtPlot::xBottom, 45);
37      updateAxes();
38
39      d_barChartItem->attach(this);
40
41      insertLegend(new QwtLegend());
42
43      populate(frequencias, count);
44
45      setUpdatesEnabled(true);
46      setAxisAutoScale(true);
47
48      QwtPlotGrid *grid = new QwtPlotGrid();
49      grid->attach(this);
50      grid->setPen(QPen(QColor(200, 200, 200)));
51
52      replot();
53  }
54
55  void GraficoBarrasQualQuantDiscreto::populate(
56      const vector<int> &frequencias,
57      const int count)
58  {
59      QwtColumnSymbol *symbol = new QwtColumnSymbol(QwtColumnSymbol::Box);
60      symbol->setLineWidth(1);
61      symbol->setFrameStyle(QwtColumnSymbol::Raised);
62      symbol->setPalette(QPalette(Qt::blue));
63
64      d_barChartItem->setSymbol(symbol);
65
66      QVector<double> series;
67      for (int i = 0; i < count; i++)
68      {
69          series += frequencias[i];
70      }
71
72      d_barChartItem->setSamples(series);
73
74  }
```

Listagem B.10: grafico_barras_qual_quant_discreto.cpp

```
1  #ifndef GRAFICO_BARRAS_QUAL_QUANT_DISCRETO
2  #define GRAFICO_BARRAS_QUAL_QUANT_DISCRETO
3
4  #include <qwt_plot.h>
5  #include <qwt_plot_renderer.h>
6  #include <qwt_plot_canvas.h>
7  #include <qwt_plot_barchart.h>
8  #include <qwt_column_symbol.h>
9  #include <qwt_plot_layout.h>
10 #include <qwt_legend.h>
11 #include "escala_nomes.h"
12
13 #include <qdatetime.h>
14
15 using namespace std;
16
17 class QwtPlotBarChart;
18 class GraficoBarrasQualQuantDiscreto: public QwtPlot
19 {
20     Q_OBJECT
21
22 public:
23     GraficoBarrasQualQuantDiscreto(
24         const vector<string> &categorias,
25         const vector<int> &frequencias,
26         const int count,
27         QWidget *parent = NULL);
28
29 private:
30     void populate(
31         const vector<int> &frequencias,
32         const int count);
```

```
33      QwtPlotBarChart *d_barChartItem;
34  };
35
36  #endif
```

Listagem B.11: grafico_barras_qual_quant_discreto.h

```
1   #include "histograma_quant_continuo.h"
2
3   #include <qwt_plot_grid.h>
4
5   class Histogram : public QwtPlotHistogram
6   {
7   public:
8       Histogram(const QString &, const QColor &);
9
10      void setColor(const QColor &);
11      void setValues(
12          const vector<int> &valores,
13          const vector<pair<double, double>> &intervalos,
14          const int numVal);
15  };
16
17  Histogram::Histogram(const QString &title, const QColor &symbolColor):
18  QwtPlotHistogram(title)
19  {
20      setStyle(QwtPlotHistogram::Columns);
21      setColor(symbolColor);
22  }
23
24  void Histogram::setColor(const QColor &color)
25  {
26      QColor c = color;
27      //c.setAlpha(180); //transparencia
28      setBrush(QBrush(c));
29  }
30
31  void Histogram::setValues(
32      const vector<int> &frequencias,
33      const vector<pair<double, double>> &intervalos,
34      const int numVal)
35  {
36      QVector<QwtIntervalSample> samples(numVal);
37      for (uint i = 0; i < numVal; i++)
38      {
39          QwtInterval interval(intervalos[i].first, intervalos[i].second);
40          interval.setBorderFlags(QwtInterval::ExcludeMaximum);
41
42          samples[i] = QwtIntervalSample(frequencias[i], interval);
43      }
44
45      setData(new QwtIntervalSeriesData(samples));
46  }
47
48  HistogramaQuantContinuo::HistogramaQuantContinuo(
49      const vector<int> &frequencias,
50      const vector<pair<double, double>> &intervalos,
51      const int numVal,
52      QWidget *parent):
53  QwtPlot(parent)
54  {
55      setAutoReplot(false);
56
57      QwtPlotCanvas *canvas = new QwtPlotCanvas();
58      //canvas->setPalette(Qt::gray);
59      //canvas->setBorderRadius(10);
60      setCanvas(canvas);
61
62      plotLayout()->setAlignCanvasToScales(true);
63
64      populate(frequencias, intervalos, numVal);
65
66      replot();
67  }
68
69  void HistogramaQuantContinuo::populate(
70      const vector<int> &frequencias,
71      const vector<pair<double, double>> &intervalos,
72      const int numVal)
```

```
73 {
74     QwtPlotGrid *grid = new QwtPlotGrid();
75     grid->attach(this);
76     grid->setPen(QPen(QColor(200, 200, 200)));
77
78     Histogram *histogram = new Histogram("", Qt::blue);
79
80     histogram->setValues(frequencias, intervalos, numVal);
81
82     this->detachItems(QwtPlotItem::Rtti_PlotHistogram, true);
83
84     histogram->attach(this);
85
86 }
```

Listagem B.12: histograma_quant_continuo.cpp

```
1  #ifndef _HISTOGRAMA_H_
2  #define _HISTOGRAMA_H_
3
4  #include <qwt_plot.h>
5  #include <qwt_plot_layout.h>
6  #include <qwt_plot_canvas.h>
7  #include <qwt_plot_renderer.h>
8  #include <qwt_legend.h>
9  #include <qwt_legend_label.h>
10 #include <qwt_plot_histogram.h>
11 #include <qwt_column_symbol.h>
12 #include <qwt_series_data.h>
13
14 #include <qpen.h>
15 #include <stdlib.h>
16 #include <vector>
17 #include <string>
18
19 using namespace std;
20
21 class HistogramaQuantContinuo: public QwtPlot
22 {
23     Q_OBJECT
24
25 public:
26     HistogramaQuantContinuo(
27         const vector<int> &frequencias,
28         const vector<pair<double, double>> &intervalos,
29         const int numVal,
30         QWidget *parent = NULL);
31
32 private:
33     void populate(
34         const vector<int> &frequencias,
35         const vector<pair<double, double>> &intervalos,
36         const int numVal);
37 };
38
39 #endif
```

Listagem B.13: histograma_quant_continuo.h

```
1  #include "item_tabela_editora.h"
2
3  #include <QLocale>
4
5  ItemTabelaEditora::ItemTabelaEditora() : QStandardItem() { }
6  ItemTabelaEditora::ItemTabelaEditora(QString &data) : QStandardItem(data) {
       }
7
8  bool ItemTabelaEditora::operator< (const QStandardItem &other) const
9  {
10     bool okThis;
11     bool okOther;
12
13     double thisDbl = QLocale(QLocale::Portuguese, QLocale::Brazil).toDouble
           (this->text(), &okThis);
14     double otherDbl = QLocale(QLocale::Portuguese, QLocale::Brazil).
           toDouble(other.text(), &okOther);
```

```
15
16      //Se os dois forem double;
17      if (okThis && okOther)
18      {
19          return thisDbl < otherDbl;
20      }
21
22      return QStandardItem::operator<(other);
23  }
```

Listagem B.14: item_tabela_editora.cpp

```
1  #ifndef ITEMTABELAEDITORA_H
2  #define ITEMTABELAEDITORA_H
3
4  #include <QStandardItem>
5
6  class ItemTabelaEditora : public QStandardItem
7  {
8  public:
9      ItemTabelaEditora();
10     ItemTabelaEditora(QString &data);
11     bool operator< (const QStandardItem &other) const;
12     bool operator<(const ItemTabelaEditora &other) const;
13 };
14 #endif
```

Listagem B.15: item_tabela_editora.h

```
1  #include "populador.h"
2
3  #include "item_tabela_editora.h"
4  #include "utilidades.h"
5
6  #include <QStandardItemModel>
7  #include <QCoreApplication>
8
9  #include <cassert>
10
11 void Populador::populeTabela(
12     QTableView *tabela,
13     const vector<string> &atributos,
14     const vector<vector<string>> &linhas)
15 {
16     const size_t totalLinhas = linhas.size();
17     const size_t totalColunas = atributos.size();
18
19     QProgressDialog *progress = new QProgressDialog(QString::fromLatin1("
           Preenchendo tabela de edição..."), "", 0, 100, NULL);
20     progress->setWindowTitle(QString::fromLatin1("Aguarde"));
21     progress->setWindowModality(Qt::ApplicationModal);
22     progress->setWindowFlags(progress->windowFlags() & ~Qt::
           WindowContextHelpButtonHint & ~Qt::WindowCloseButtonHint | Qt::
           MSWindowsFixedSizeDialogHint);
23     progress->setCancelButton(NULL);
24     progress->setValue(0);
25     progress->show();
26
27     const int taxaNotificacao = totalLinhas <= 10 ? 10 : totalLinhas / 10;
28     size_t i = 0;
29
30     QStandardItemModel *model = (QStandardItemModel*)tabela->model();
31
32     model->setColumnCount(totalColunas);
33     model->setRowCount(totalLinhas);
34
35     ItemTabelaEditora *item;
36
37     tabela->setUpdatesEnabled(false);
38     tabela->blockSignals(true);
39     model->blockSignals(true);
40
41     size_t coluna = 0;
42     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
43         citAtributo != atributos.cend();
44         citAtributo++, coluna++)
```

```
45      {
46          item = new ItemTabelaEditora(QString::fromLatin1(citAtributo->c_str
                ()));
47          model->setHorizontalHeaderItem(coluna, item);
48      }
49
50      size_t linha = 0;
51      coluna = 0;
52      for (vector<vector<string>>::const_iterator citLinha = linhas.cbegin();
53          citLinha != linhas.cend();
54          citLinha++, linha++)
55      {
56          coluna = 0;
57          for (vector<string>::const_iterator citColuna = citLinha->cbegin();
58          citColuna != citLinha->cend();
59          citColuna++, coluna++)
60          {
61              item = new ItemTabelaEditora(QString::fromLatin1(citColuna->
                    c_str()));
62              item->setData(Qt::AlignRight, Qt::TextAlignmentRole);
63              model->setItem(linha, coluna, item);
64          }
65
66          i++;
67          if ((i % taxaNotificacao) == 0)
68          {
69              progress->setValue(min((int)(i * 1.0 / totalLinhas * 100), 99))
                    ;
70              QCoreApplication::processEvents(QEventLoop::
                    ExcludeUserInputEvents);
71          }
72      }
73
74      tabela->blockSignals(false);
75      tabela->setUpdatesEnabled(true);
76      model->blockSignals(false);
77      emit model->layoutChanged();
78
79      //tabela->resizeColumnsToContents(); lento demais
80
81      delete progress;
82  }
83
84  void Populador::populeColunaTabelaEditora(
85      const map<size_t, string> &valoresColuna,
86      const int indiceColuna,
87      QTableView *tabela,
88      const string atributo)
89  {
90      QStandardItemModel *model = (QStandardItemModel*)tabela->model();
91
92      assert(model->rowCount() == valoresColuna.size());
93
94      tabela->setUpdatesEnabled(false);
95      tabela->blockSignals(true);
96      model->blockSignals(true);
97
98      ItemTabelaEditora *item;
99
100     if (!atributo.empty())
101     {
102         item = new ItemTabelaEditora(QString::fromLatin1(atributo.c_str()))
                ;
103         model->setHorizontalHeaderItem(indiceColuna, item);
104     }
105
106     const int totalLinhas = model->rowCount();
107
108     for (int linha = 0; linha < totalLinhas; linha++)
109     {
110         const size_t indiceRegistro = model->item(linha, 0)->text().
                toLatin1().toInt();
111
112         item = (ItemTabelaEditora*)(model->item(linha, indiceColuna));
113         if (!item)
114         {
115             item = new ItemTabelaEditora();
116             item->setData(Qt::AlignRight, Qt::TextAlignmentRole);
117             model->setItem(linha, indiceColuna, item);
118         }
119
120         item->setText(QString::fromLatin1(valoresColuna.at(indiceRegistro).
```

```
                c_str()));
121     }
122
123     tabela->setUpdatesEnabled(true);
124     tabela->blockSignals(false);
125     model->blockSignals(false);
126     emit model->layoutChanged();
127
128     //tabela->resizeColumnToContents(indiceColuna); lento demais
129 }
130
131 void Populador::populeLinhaTabela(
132     const vector<string> &valoresLinha,
133     const int linhaTabela,
134     QTableView *tabela)
135 {
136     QStandardItemModel *model = (QStandardItemModel*)tabela->model();
137
138     assert(valoresLinha.size() == model->columnCount());
139
140     ItemTabelaEditora *item;
141
142     tabela->setUpdatesEnabled(false);
143     tabela->blockSignals(true);
144     model->blockSignals(true);
145
146     size_t coluna = 0;
147     for (vector<string>::const_iterator citValor = valoresLinha.cbegin();
148          citValor != valoresLinha.cend();
149          citValor++, coluna++)
150     {
151         item = new ItemTabelaEditora(QString::fromLatin1(citValor->c_str())
                );
152         item->setData(Qt::AlignRight, Qt::TextAlignmentRole);
153         model->setItem(linhaTabela, coluna, item);
154     }
155
156     tabela->setUpdatesEnabled(true);
157     tabela->blockSignals(false);
158     model->blockSignals(false);
159     emit model->layoutChanged();
160
161     //tabela->->resizeColumnsToContents(); lento demais
162 }
163
164 void Populador::populeCelulasEspecificasTabela(
165     const vector<pair<size_t, size_t>> &celulasIndices,
166     const vector<string> &valores,
167     QTableView *tabela)
168 {
169     QStandardItemModel *model = (QStandardItemModel*)tabela->model();
170
171     assert(celulasIndices.size() == valores.size());
172
173     ItemTabelaEditora *item;
174
175     tabela->setUpdatesEnabled(false);
176     tabela->blockSignals(true);
177     model->blockSignals(true);
178
179     vector<string>::const_iterator citValor = valores.cbegin();
180     for (vector<pair<size_t, size_t>>::const_iterator citCelulaIndice =
           celulasIndices.cbegin();
181          citCelulaIndice != celulasIndices.cend();
182          citCelulaIndice++, citValor++)
183     {
184         const size_t linha = citCelulaIndice->first;
185         const size_t coluna = citCelulaIndice->second;
186         const string valorStr = *citValor;
187
188         item = new ItemTabelaEditora(QString::fromLatin1(valorStr.c_str()))
                ;
189         item->setData(Qt::AlignRight, Qt::TextAlignmentRole);
190         model->setItem(linha, coluna, item);
191     }
192
193     tabela->setUpdatesEnabled(true);
194     tabela->blockSignals(false);
195     model->blockSignals(false);
196     emit model->layoutChanged();
197
198     //tabela->->resizeColumnsToContents(); lento demais
```

```
199 }
200
201 void Populador::populeComboBox(
202     QComboBox *combo,
203     const vector<string> &valores)
204 {
205     combo->blockSignals(true);
206
207     combo->clear();
208
209     for (vector<string>::const_iterator citValor = valores.cbegin();
210         citValor != valores.cend();
211         citValor++)
212     {
213         combo->addItem(QString::fromLatin1(citValor->c_str()));
214     }
215
216     combo->blockSignals(false);
217 }
218
219 void Populador::populeListWidget(
220     QListWidget *list,
221     const vector<string> &valores)
222 {
223     list->blockSignals(true);
224
225     list->clear();
226
227     QListWidgetItem *item;
228     for (vector<string>::const_iterator citValor = valores.cbegin();
229         citValor != valores.cend();
230         citValor++)
231     {
232         item = new QListWidgetItem();
233         item->setData(Qt::DisplayRole, QString::fromLatin1(citValor->c_str
                ()));
234         item->setData(Qt::CheckStateRole, Qt::Unchecked);
235
236         list->addItem(item);
237     }
238
239     list->blockSignals(false);
240 }
241
242 void Populador::populeArvoreTipos(
243     QTreeWidget *arvore,
244     const vector<string> &atributos,
245     const vector<TipoDado_AT> &tipos)
246 {
247     arvore->blockSignals(true);
248
249     assert(atributos.size() == tipos.size());
250
251     arvore->clear();
252
253     QTreeWidgetItem *nodoQuantContinuo = new QTreeWidgetItem(arvore);
254     nodoQuantContinuo->setText(0, QString::fromLatin1("Quantitativo
            Contínuo"));
255     arvore->addTopLevelItem(nodoQuantContinuo);
256
257     QTreeWidgetItem *nodoQuantDiscreto = new QTreeWidgetItem(arvore);
258     nodoQuantDiscreto->setText(0, QString::fromLatin1("Quantitativo
            Discreto"));
259     arvore->addTopLevelItem(nodoQuantDiscreto);
260
261     QTreeWidgetItem *nodoQualitativo = new QTreeWidgetItem(arvore);
262     nodoQualitativo->setText(0, QString::fromLatin1("Qualitativo"));
263     arvore->addTopLevelItem(nodoQualitativo);
264
265     QTreeWidgetItem *child;
266
267     vector<string>::const_iterator citAtributo = atributos.cbegin();
268     for (vector<TipoDado_AT>::const_iterator citTipo = tipos.cbegin();
269         citTipo != tipos.cend();
270         citTipo++, citAtributo++)
271     {
272         if (*citTipo == QUALITATIVO)
273         {
274             child = new QTreeWidgetItem(nodoQualitativo);
275         }
276         else if (*citTipo == QUANT_CONTINUO)
277         {
```

```
278                 child = new QTreeWidgetItem(nodoQuantContinuo);
279             }
280             else //if (*citTipo == QUANT_DISCRETO)
281             {
282                 child = new QTreeWidgetItem(nodoQuantDiscreto);
283             }
284
285             child->setText(0, QString::fromLatin1(citAtributo->c_str()));
286         }
287
288         arvore->blockSignals(false);
289     }
290
291     void Populador::populeTabelaEstatistica(
292         QTableWidget *tabela,
293         const vector<pair<string, double>> &informacoes)
294     {
295         QAbstractItemModel *model = tabela->model();
296
297         tabela->setUpdatesEnabled(false);
298         tabela->blockSignals(true);
299         model->blockSignals(true);
300
301         tabela->setRowCount(0);
302         tabela->setRowCount(informacoes.size());
303         tabela->setColumnCount(2);
304
305         size_t linha = 0;
306         for (vector<pair<string, double>>::const_iterator citInf = informacoes.
                cbegin();
307             citInf != informacoes.cend();
308             citInf++, linha++)
309         {
310             QTableWidgetItem *item = new QTableWidgetItem(QString::fromLatin1(
                    citInf->first.c_str()));
311             tabela->setItem(linha, 0, item);
312
313             item = new QTableWidgetItem(QString::fromLatin1(Utilidades::
                    dblToStdStr(citInf->second).c_str()));
314             tabela->setItem(linha, 1, item);
315         }
316
317         tabela->setUpdatesEnabled(true);
318         tabela->blockSignals(false);
319         model->blockSignals(false);
320         emit model->layoutChanged();
321     }
```

Listagem B.16: populador.cpp

```
1  #ifndef POPULADOR_H
2  #define POPULADOR_H
3
4  #include <QTableView>
5  #include <QTableWidget>
6  #include <QComboBox>
7  #include <QListWidget>
8  #include <QTreeWidget>
9  #include <string>
10
11 #include "enums_at.h"
12
13 using namespace std;
14
15 class Populador
16 {
17 public:
18     static void populeTabela(
19         QTableView *tabela,
20         const vector<string> &atributos,
21         const vector<vector<string>> &linhas);
22
23     static void populeColunaTabelaEditora(
24         const map<size_t, string> &valoresColuna,
25         const int indiceColuna,
26         QTableView *tabela,
27         const string nome = "");
28
29     static void populeLinhaTabela(
```

```
30          const vector<string> &valoresLinha,
31          const int linhaTabela,
32          QTableView *tabela);
33
34      static void populeCelulasEspecificasTabela(
35          const vector<pair<size_t, size_t>> &celulasIndices,
36          const vector<string> &valores,
37          QTableView *tabela);
38
39      static void populeComboBox(
40          QComboBox *combo,
41          const vector<string> &valores);
42
43      static void populeListWidget(
44          QListWidget *list,
45          const vector<string> &valores);
46
47      static void populeArvoreTipos(
48          QTreeWidget *arvore,
49          const vector<string> &atributos,
50          const vector<TipoDado_AT> &tipos);
51
52      static void populeTabelaEstatistica(
53          QTableWidget *tabela,
54          const vector<pair<string, double>> &informacoes);
55  };
56  #endif
```

Listagem B.17: populador.h

```
1   #include "diagrama_caixa.h"
2
3   DiagramaCaixa::DiagramaCaixa(QWidget *parent)
4       : QCustomPlot(parent)
5   {
6       _caixa = new QCPStatisticalBox(xAxis, yAxis);
7       _caixa->setKey(1);
8
9       addPlottable(_caixa);
10
11      xAxis->setSubTickCount(0);
12      xAxis->setTickLength(0, 4);
13      //xAxis->setTickLabelRotation(20);
14      xAxis->setAutoTicks(false);
15      xAxis->setAutoTickLabels(false);
16      xAxis->setTickVector(QVector<double>() << 1);
17
18      rescaleAxes();
19      xAxis->scaleRange(2, xAxis->range().center()); //largura da caixa
20      setInteractions(QCP::iRangeDrag | QCP::iRangeZoom);
21
22      replot();
23  }
24
25  DiagramaCaixa::~DiagramaCaixa()
26  {
27      const void * address = static_cast<const void*>(&(*this));
28      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
29  }
30
31  void DiagramaCaixa::setDados(
32      const string &atributo,
33      const double maximo,
34      const double whiskerSuperior,
35      const double quartilSuperior,
36      const double mediana,
37      const double quartilInferior,
38      const double whiskerInferior,
39      const double minimo,
40      const vector<double> &outliers)
41  {
42      const double diferenca = ((maximo - minimo) != 0) ? maximo - minimo :
            100;
43
44      _caixa->setMaximum(whiskerSuperior);
45      _caixa->setUpperQuartile(quartilSuperior);
46      _caixa->setMedian(mediana);
47      _caixa->setLowerQuartile(quartilInferior);
48      _caixa->setMinimum(whiskerInferior);
```

```
49      _caixa->setOutliers(QVector<double>::fromStdVector(outliers));
50
51      xAxis->setTickVectorLabels(QVector<QString>() << QString::fromLatin1(
            atributo.c_str()));
52      yAxis->setRange(whiskerInferior - 0.05 * diferenca, whiskerSuperior +
            0.05 * diferenca);
53
54      replot();
55  }
```

Listagem B.18: diagrama_caixa.cpp

```
1  #ifndef DIAGRAMA_CAIXA_H
2  #define DIAGRAMA_CAIXA_H
3
4  #include <qcustomplot.h>
5
6  #include <vector>
7  #include <string>
8
9  using namespace std;
10
11 class DiagramaCaixa : public QCustomPlot
12 {
13 public:
14     DiagramaCaixa(QWidget *parent);
15     ~DiagramaCaixa();
16
17     void DiagramaCaixa::setDados(
18         const string &atributo,
19         const double max,
20         const double whiskerSuperior,
21         const double quartilSuperior,
22         const double mediana,
23         const double quartilInferior,
24         const double whiskerInferior,
25         const double min,
26         const vector<double> &outliers);
27
28 private:
29     QCPStatisticalBox *_caixa;
30 };
31
32 #endif // DIAGRAMA_CAIXA_H
```

Listagem B.19: diagrama_caixa.h

```
1  #include "diagrama_scatter.h"
2
3  #include "utilidades.h"
4
5  #include <qwt_plot_grid.h>
6  #include <qwt_plot_magnifier.h>
7  #include <qwt_plot_panner.h>
8
9  DiagramaDispersao::DiagramaDispersao(bool tresVariaveis, QWidget *parent) :
       QwtPlot(parent), _tituloLegenda(NULL)
10 {
11     setAutoReplot(false);
12
13     if (tresVariaveis)
14     {
15         _tituloLegenda = new QwtPlotTextLabel();
16         _tituloLegenda->attach(this);
17
18
19         insertLegend(new QwtLegend(), QwtPlot::RightLegend);
20     }
21
22     setCanvasBackground(Qt::white);
23
24     QList<QString> a;
25     a.push_back("");
26
27     setTotalCurvas(a.size(), a);
28
```

```
29      QwtPlotGrid *grid = new QwtPlotGrid();
30      grid->attach(this);
31      grid->setPen(QPen(QColor(200, 200, 200)));
32
33      // panning with the left mouse button
34      (void )new QwtPlotPanner( canvas() );
35      // zoom in/out with the wheel
36      QwtPlotMagnifier *magnifier = new QwtPlotMagnifier( canvas() );
37      magnifier->setMouseButton(Qt::NoButton);
38      magnifier->setWheelFactor(-1.1);
39  }
40
41  void DiagramaDispersao::setTituloLegenda(QString titulo)
42  {
43      if (_tituloLegenda)
44      {
45          QwtText title(titulo);
46          title.setRenderFlags( Qt::AlignRight | Qt::AlignTop );
47          title.setFont(axisTitle(QwtPlot::xBottom).font());
48          _tituloLegenda->setText(title);
49      }
50  }
51
52  DiagramaDispersao::~DiagramaDispersao()
53  {
54      const void * address = static_cast<const void*>(&(*this));
55      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
56
57      clearPoints();
58
59      for (int i = 0; i < curvas.size(); i++)
60      {
61          delete curvas[i];
62      }
63
64      curvas.clear();
65
66      delete _tituloLegenda;
67  }
68
69  void DiagramaDispersao::setTotalCurvas(int total, QList<QString> atributos)
70  {
71      clearPoints();
72
73      for (int i = 0; i < curvas.size(); i++)
74      {
75          delete curvas[i];
76      }
77
78      curvas.clear();
79
80
81      if (total < 0 || atributos.size() != total)
82      {
83          return;
84      }
85
86      QwtPointSeriesData *dadosCurva;
87      QwtPlotCurve *novaCurva;
88
89      for (int i = 0; i < total; i++)
90      {
91          dadosCurva = new QwtPointSeriesData();
92
93          novaCurva = new QwtPlotCurve(atributos[i]);
94          novaCurva->setData(dadosCurva);
95          novaCurva->setRenderHint(QwtPlotItem::RenderAntialiased);
96          novaCurva->setPen(QPen(QColor(Utilidades::coresDistintas[i % 56].
                c_str())));
97          novaCurva->setStyle(QwtPlotCurve::NoCurve);
98          novaCurva->setSymbol(new QwtSymbol(QwtSymbol::XCross, Qt::NoBrush,
                QPen(QColor(Utilidades::coresDistintas[i % 56].c_str())),
                QSize(4, 4)));
99          novaCurva->attach(this);
100
101         curvas.append(novaCurva);
102     }
103 }
104
105 void DiagramaDispersao::clearPoints()
106 {
107     QwtPlotCurve *curva;
```

```
108     QwtArraySeriesData<QPointF> *data;
109
110     for (int i = 0; i < curvas.size(); i++)
111     {
112         curva = curvas[i];
113         data = dynamic_cast<QwtPointSeriesData*>(curva->data());
114         data->clear();
115     }
116 }
117
118 void DiagramaDispersao::appendPoint(const QPointF &point, size_t
        indiceCurva)
119 {
120     if (indiceCurva > curvas.size()){return;}
121
122     QwtPlotCurve *curva = curvas[indiceCurva];
123
124     QwtPointSeriesData *data = dynamic_cast<QwtPointSeriesData*>(curva->
            data());
125
126     data->append(point);
127 }
```

Listagem B.20: diagrama_scatter.cpp

```
1  #ifndef DIAGRAMA_DISPERSAO_H
2  #define DIAGRAMA_DISPERSAO_H
3
4  #include <qwt_plot.h>
5  #include <qwt_plot_marker.h>
6  #include <qwt_plot_curve.h>
7  #include <qwt_legend.h>
8  #include <qwt_series_data.h>
9  #include <qwt_text.h>
10 #include <qwt_math.h>
11 #include <qwt_point_data.h>
12 #include <qwt_symbol.h>
13 #include <qwt_plot_textlabel.h>
14
15 using namespace std;
16
17 class DiagramaDispersao : public QwtPlot
18 {
19 private:
20     QList<QwtPlotCurve*> curvas;
21     QwtPlotTextLabel *_tituloLegenda;
22
23
24 public:
25     DiagramaDispersao(bool tresVariaveis = false, QWidget *parent = NULL);
26     ~DiagramaDispersao();
27     void setTotalCurvas(int total, QList<QString> atributos);
28     void clearPoints();
29     void appendPoint(const QPointF &, size_t indiceCurva = 0);
30     void setTituloLegenda(QString titulo);
31 };
32 #endif // DIAGRAMA_DISPERSAO_H
```

Listagem B.21: diagrama_scatter.h

```
1  #include "thread_worker_coletar_estatisticas.h"
2
3  #include "coletor_estatisticas.h"
4
5  ThreadWorkerColetarEstatisticas::ThreadWorkerColetarEstatisticas(
6          const vector<string> &atributos,
7          const vector<TipoDado_AT> &tipos,
8          const vector<vector<ValorBase *const>> &dados,
9          QWidget *parent)
10         : _atributos(atributos), _tipos(tipos), _dados(dados)
11 {
12     this->paradaManual = false;
13     if (parent)
14     {
15         QObject::connect(this, SIGNAL(fimColetaEstatisticas(const vector<
               string>*)), parent, SLOT(fimColetaEstatisticas(const vector<
               string>*)));
```

```
16      }
17  }
18
19  ThreadWorkerColetarEstatisticas::ThreadWorkerColetarEstatisticas(
20          const vector<string> &atributos,
21          QWidget *parent)
22          : _atributos(atributos)
23  {
24      this->paradaManual = false;
25      if (parent)
26      {
27          QObject::connect(this, SIGNAL(fimColetaEstatisticas(const vector<
                string>*)), parent, SLOT(fimColetaEstatisticas(const vector<
                string>*)));
28      }
29  }
30
31  void ThreadWorkerColetarEstatisticas::pararExecucao()
32  {
33      this->paradaManual = true;
34  }
35
36  void ThreadWorkerColetarEstatisticas::executarCodigo()
37  {
38      if (!_dados.empty())
39      {
40          TabelaDeRegistros::obterInstancia()->inicialize(_dados, _atributos,
                _tipos, this);
41      }
42
43      ColetorEstatisticas::obterInstancia()->coletarEstatisticasDosAtributos(
            _atributos, this);
44
45      const vector<string> *atributos = new vector<string>(_atributos);
46      emit fimColetaEstatisticas(atributos);
47      emit finalizouExecucao();
48  }
49
50  void ThreadWorkerColetarEstatisticas::Notificar(int *porcentagem)
51  {
52      emit updateProgresso(porcentagem);
53  }
```

Listagem B.22: thread_worker_coletar_estatisticas.cpp

```
1   #ifndef THREADWORKERCOLETARESTATISTICAS_H
2   #define THREADWORKERCOLETARESTATISTICAS_H
3
4   #include "thread_worker.h"
5
6   #include <QObject>
7   #include <QWidget>
8   #include <QThread>
9   #include <QMutex>
10  #include <QWaitCondition>
11
12  #include "enums_at.h"
13  #include "valor_base.h"
14
15  #include <vector>
16  #include <string>
17
18  using namespace std;
19
20  class ThreadWorkerColetarEstatisticas :
21      public ThreadWorker
22  {
23      Q_OBJECT
24
25  private:
26      vector<string> _atributos;
27      vector<TipoDado_AT> _tipos;
28      vector<vector<ValorBase *const>> _dados;
29
30  protected:
31      bool paradaManual;
32  public:
33      ThreadWorkerColetarEstatisticas(
34          const vector<string> &atributos,
```

```
35          const vector<TipoDado_AT> &tipos,
36          const vector<vector<ValorBase *const>> &dados,
37          QWidget *parent = 0);
38
39      ThreadWorkerColetarEstatisticas(
40          const vector<string> &atributos,
41          QWidget *parent = 0);
42
43      QMutex sync;
44      QWaitCondition pauseCond;
45      bool stop;
46      bool started;
47  #ifndef TCC_RODOLFO
48      void Notificar(StatusThread *status){};
49  #endif
50      void Notificar(int *porcentagem);
51
52      virtual void executarCodigo();
53      virtual void pararExecucao();
54
55  signals:
56      void fimColetaEstatisticas(const vector<string> *atributos);
57  };
58
59  #endif
```

Listagem B.23: thread_worker_coletar_estatisticas.h

B.1.1 Telas

```
1  #include "dialog_edicao_dados.h"
2
3  #include <QDebug>
4
5  #include "widget_tabela_editora.h"
6
7  DialogEdicaoDados::DialogEdicaoDados(const vector<size_t> &indicesRegistros
       , QWidget *parent) : QDialog(parent)
8  {
9      ui.setupUi(this);
10
11     WidgetTabelaEditora *widgetTabelaEditora = new WidgetTabelaEditora(
           indicesRegistros, parent);
12     this->layout()->addWidget(widgetTabelaEditora);
13
14     setWindowFlags((windowFlags() | Qt::WindowMaximizeButtonHint) & ~Qt::
           WindowContextHelpButtonHint);
15 }
16
17 DialogEdicaoDados::~DialogEdicaoDados()
18 {
19     const void * address = static_cast<const void*>(&(*this));
20     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
21 }
```

Listagem B.24: dialog_edicao_dados.cpp

```
1  #ifndef DIALOG_EDICAO_DADOS_H
2  #define DIALOG_EDICAO_DADOS_H
3
4  #include "ui_dialog_edicao_dados.h"
5
6  using namespace std;
7
8  class DialogEdicaoDados : public QDialog
9  {
10     Q_OBJECT
11
12 public:
13     DialogEdicaoDados(const vector<size_t> &indicesRegistros, QWidget *
           parent = 0);
```

```
14      ~DialogEdicaoDados();
15
16  private:
17      Ui::DialogEdicaoDados ui;
18  };
19
20  #endif // DIALOG_EDICAO_DADOS_H
```

Listagem B.25: dialog_edicao_dados.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogEdicaoDados</class>
4   <widget class="QDialog" name="DialogEdicaoDados">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>842</width>
10     <height>491</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>Edição dos Dados</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout_2">
17    <property name="leftMargin">
18     <number>0</number>
19    </property>
20    <property name="topMargin">
21     <number>0</number>
22    </property>
23    <property name="rightMargin">
24     <number>0</number>
25    </property>
26    <property name="bottomMargin">
27     <number>0</number>
28    </property>
29   </layout>
30  </widget>
31  <layoutdefault spacing="6" margin="11"/>
32  <resources/>
33  <connections/>
34 </ui>
```

Listagem B.26: dialog_edicao_dados.ui

```
1  #include "dialog_escolher_distribuicao.h"
2
3  #include <QDebug>
4
5  DialogEscolherDistribuicao::DialogEscolherDistribuicao(
6      const vector<string> &atributos,
7      const vector<vector<size_t>> &indicesValidosPorAtributo,
8      const WidgetEscolherDP::Opcao opcao,
9      QWidget *parent)
10     : QDialog(parent)
11 {
12     inicialize(atributos, indicesValidosPorAtributo, opcao);
13 }
14
15 DialogEscolherDistribuicao::DialogEscolherDistribuicao(
16     const string &atributo,
17     const vector<size_t> &indicesValidos,
18     const WidgetEscolherDP::Opcao opcao,
19     QWidget *parent)
20     : QDialog(parent)
21 {
22     vector<string> atributos = vector<string>();
23     atributos.push_back(atributo);
24
25     vector<vector<size_t>> indicesValidosPorAtributo = vector<vector<size_t
           >>();
26     indicesValidosPorAtributo.push_back(indicesValidos);
27
28     inicialize(atributos, indicesValidosPorAtributo, opcao);
```

```
29 }
30
31 DialogEscolherDistribuicao::~DialogEscolherDistribuicao()
32 {
33     const void * address = static_cast<const void*>(&(*this));
34     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
35 }
36
37 void DialogEscolherDistribuicao::inicialize(
38     const vector<string> &atributos,
39     const vector<vector<size_t>> &indicesValidosPorAtributo,
40     const WidgetEscolherDP::Opcao opcao)
41 {
42     ui.setupUi(this);
43
44     _widgetEscolherDP = new WidgetEscolherDP(atributos,
          indicesValidosPorAtributo, opcao, this);
45     this->layout()->addWidget(_widgetEscolherDP);
46
47     setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
48 }
```

Listagem B.27: dialog_escolher_distribuicao.cpp

```
1  #ifndef DIAOG_ESCOLHER_DISTRIBUICAO_H
2  #define DIAOG_ESCOLHER_DISTRIBUICAO_H
3
4  #include "ui_dialog_escolher_distribuicao.h"
5
6  #include "widget_escolher_dp.h"
7
8  using namespace std;
9
10 class DialogEscolherDistribuicao : public QDialog
11 {
12     Q_OBJECT
13
14 public:
15     WidgetEscolherDP *_widgetEscolherDP;
16
17     DialogEscolherDistribuicao(
18         const vector<string> &atributos,
19         const vector<vector<size_t>> &indicesValidosPorAtributo,
20         const WidgetEscolherDP::Opcao opcao,
21         QWidget *parent = 0);
22     DialogEscolherDistribuicao(
23         const string &atributo,
24         const vector<size_t> &indicesValidos,
25         const WidgetEscolherDP::Opcao opcao,
26         QWidget *parent = 0);
27     ~DialogEscolherDistribuicao();
28
29     void inicialize(
30         const vector<string> &atributos,
31         const vector<vector<size_t>> &indicesValidosPorAtributo,
32         const WidgetEscolherDP::Opcao opcao);
33
34 private:
35     Ui::DialogEscolherDistribuicao ui;
36 };
37
38 #endif // DIAOG_ESCOLHER_DISTRIBUICAO_H
```

Listagem B.28: dialog_escolher_distribuicao.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogEscolherDistribuicao</class>
4   <widget class="QDialog" name="DialogEscolherDistribuicao">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>609</width>
10     <height>396</height>
11    </rect>
```

```
12    </property>
13    <property name="windowTitle">
14     <string>Escolher Distribuicão</string>
15    </property>
16    <layout class="QVBoxLayout" name="verticalLayout_2"/>
17   </widget>
18   <layoutdefault spacing="6" margin="11"/>
19   <resources/>
20   <connections/>
21  </ui>
```

Listagem B.29: dialog_escolher_distribuicao.ui

```
1  #include "dialog_escolher_RLS.h"
2
3  #include "populador.h"
4
5  #include <QDebug>
6
7  DialogEscolherRLS::DialogEscolherRLS(const vector<string> &atributosDep,
       const vector<string> &todosAtributos, const vector<vector<size_t>> &
       indicesValidosPorAtributo, QWidget *parent)
8      : QDialog(parent)
9  {
10     inicialize(atributosDep, todosAtributos, indicesValidosPorAtributo);
11 }
12
13 DialogEscolherRLS::DialogEscolherRLS(const string &atributoDep, const
       vector<string> &todosAtributos, const vector<vector<size_t>> &
       indicesValidosPorAtributo, QWidget *parent)
14     : QDialog(parent)
15 {
16     vector<string> atributosDep = vector<string>();
17     atributosDep.push_back(atributoDep);
18
19     inicialize(atributosDep, todosAtributos, indicesValidosPorAtributo);
20 }
21
22 DialogEscolherRLS::~DialogEscolherRLS()
23 {
24     const void * address = static_cast<const void*>(&(*this));
25     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
26 }
27
28 void DialogEscolherRLS::inicialize(const vector<string> &atributosDep,
       const vector<string> &todosAtributos, const vector<vector<size_t>> &
       indicesValidosPorAtributo)
29 {
30     ui.setupUi(this);
31
32     _atributosDep = atributosDep;
33
34     Populador::populeComboBox(ui.comboBoxAtributos, _atributosDep);
35
36     if (_atributosDep.size() == 1)
37     {
38         ui.labelAtributo->setVisible(false);
39         ui.comboBoxAtributos->setVisible(false);
40         ui.horizontalSpacer->changeSize(0,0, QSizePolicy::Fixed,
               QSizePolicy::Fixed);
41     }
42
43     for (vector<string>::const_iterator citAtributoDep = _atributosDep.
           cbegin();
44          citAtributoDep != _atributosDep.cend();
45          citAtributoDep++)
46     {
47         const string &atributoDep = *citAtributoDep;
48
49         WidgetRegressaoLinearSimples *widgetRegressaoLinearSimples = new
               WidgetRegressaoLinearSimples(atributoDep, todosAtributos,
               indicesValidosPorAtributo);
50         _widgetsRegressaoLinearSimples.push_back(
               widgetRegressaoLinearSimples);
51
52         ui.stackedWidget->addWidget(widgetRegressaoLinearSimples);
53     }
54
55     ui.comboBoxAtributos->setCurrentIndex(0);
```

```
56      ui.stackedWidget->setCurrentIndex(0);
57
58      connect(ui.comboBoxAtributos, SIGNAL(currentIndexChanged(int)), ui.
            stackedWidget, SLOT(setCurrentIndex(int)));
59      connect(ui.pushButtonCancelar, SIGNAL(clicked()), this, SLOT(reject()))
            ;
60      connect(ui.pushButtonGerar, SIGNAL(clicked()), this, SLOT(gerarClick())
            );
61
62      setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
63  }
64
65  void DialogEscolherRLS::gerarClick()
66  {
67      vector<const RegressaoLinearSimples> regressoes = vector<const
            RegressaoLinearSimples>();
68      vector<string> atributosIndep = vector<string>();
69
70      for (vector<WidgetRegressaoLinearSimples*>::const_iterator cit =
            _widgetsRegressaoLinearSimples.cbegin();
71          cit != _widgetsRegressaoLinearSimples.cend();
72          cit++)
73      {
74          const WidgetRegressaoLinearSimples *widgetRegressaoLinearSimples =
                *cit;
75
76          regressoes.push_back(widgetRegressaoLinearSimples->
                regressaoLinearSimples());
77          atributosIndep.push_back(widgetRegressaoLinearSimples->
                atributoIndep());
78      }
79
80      emit regressoesEscolhidas(atributosIndep, _atributosDep, regressoes);
81  }
```

Listagem B.30: dialog_escolher_RLS.cpp

```
1   #ifndef DIALOG_ESCOLHER_RLS_H
2   #define DIALOG_ESCOLHER_RLS_H
3
4   #include <QDialog>
5   #include "ui_dialog_escolher_RLS.h"
6
7   #include "widget_regressao_linear_simples.h"
8
9   #include <vector>
10
11  using namespace std;
12
13  class DialogEscolherRLS : public QDialog
14  {
15      Q_OBJECT
16
17  public:
18      DialogEscolherRLS(const vector<string> &atributosDep, const vector<
            string> &todosAtributos, const vector<vector<size_t>> &
            indicesValidosPorAtributo, QWidget *parent = 0);
19      DialogEscolherRLS(const string &atributoDep, const vector<string> &
            todosAtributos, const vector<vector<size_t>> &
            indicesValidosPorAtributo, QWidget *parent = 0);
20      ~DialogEscolherRLS();
21
22      void inicialize(const vector<string> &atributosDep, const vector<string
            > &todosAtributos, const vector<vector<size_t>> &
            indicesValidosPorAtributo);
23
24  private:
25      Ui::DialogEscolherRLS ui;
26      vector<string> _atributosDep;
27      vector<WidgetRegressaoLinearSimples*> _widgetsRegressaoLinearSimples;
28
29  private slots:
30      void gerarClick();
31
32  signals:
33      void regressoesEscolhidas(const vector<string>&, const vector<string>&,
             const vector<const RegressaoLinearSimples>&);
34  };
35
```

```
36 #endif // DIALOG_ESCOLHER_RLS_H
```

Listagem B.31: dialog_escolher_RLS.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogEscolherRLS</class>
4   <widget class="QDialog" name="DialogEscolherRLS">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>400</width>
10     <height>300</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>Escolher Regressão Linear Simples</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout">
17    <property name="leftMargin">
18     <number>9</number>
19    </property>
20    <property name="topMargin">
21     <number>9</number>
22    </property>
23    <property name="rightMargin">
24     <number>9</number>
25    </property>
26    <property name="bottomMargin">
27     <number>9</number>
28    </property>
29    <item>
30     <layout class="QHBoxLayout" name="horizontalLayout">
31      <item>
32       <widget class="QLabel" name="labelAtributo">
33        <property name="text">
34         <string>Atributo:</string>
35        </property>
36       </widget>
37      </item>
38      <item>
39       <widget class="QComboBox" name="comboBoxAtributos"/>
40      </item>
41      <item>
42       <spacer name="horizontalSpacer">
43        <property name="orientation">
44         <enum>Qt::Horizontal</enum>
45        </property>
46        <property name="sizeHint" stdset="0">
47         <size>
48          <width>40</width>
49          <height>20</height>
50         </size>
51        </property>
52       </spacer>
53      </item>
54     </layout>
55    </item>
56    <item>
57     <widget class="QStackedWidget" name="stackedWidget"/>
58    </item>
59    <item>
60     <layout class="QHBoxLayout" name="horizontalLayout_2">
61      <item>
62       <spacer name="horizontalSpacer_2">
63        <property name="orientation">
64         <enum>Qt::Horizontal</enum>
65        </property>
66        <property name="sizeHint" stdset="0">
67         <size>
68          <width>40</width>
69          <height>20</height>
70         </size>
71        </property>
72       </spacer>
73      </item>
74      <item>
75       <widget class="QPushButton" name="pushButtonGerar">
```

```
76          <property name="text">
77           <string>Gerar</string>
78          </property>
79         </widget>
80        </item>
81        <item>
82         <widget class="QPushButton" name="pushButtonCancelar">
83          <property name="text">
84           <string>Cancelar</string>
85          </property>
86         </widget>
87        </item>
88       </layout>
89      </item>
90     </layout>
91    </widget>
92    <layoutdefault spacing="6" margin="11"/>
93    <resources/>
94    <connections/>
95   </ui>
```

Listagem B.32: dialog_escolher_RLS.ui

```
1  #include "dialog_importacao_dados.h"
2
3  #include "thread_worker_coletar_estatisticas.h"
4
5  #include <QDebug>
6
7  DialogImportacaoDados::DialogImportacaoDados(QWidget *parent) :
8  DialogProgresso(parent),
9      _progress(NULL),
10
11     _dadosColunasStr(),
12     _atributos(),
13     _tipos(),
14     _diretorioArquivo(),
15     _delimitador(),
16
17     _widgetsCategoriasQual(),
18     _widgetsIntervalosQuant(),
19     _widgetTabelaEditoraTratarInterCateg(NULL),
20     _regsDescPorAtrib(),
21     _indicesOutliersPorAtrib(),
22
23     _widgetsParamDiagCaixa(),
24     _widgetsParamZscore(),
25     _widgetTabelaEditoraTratarOutliers(NULL),
26
27     _widgetTabelaEditoraTratarMissingsInvalidos(NULL)
28 {
29     ui.setupUi(this);
30
31     connect(ui.pushButtonProximo, SIGNAL(clicked()), this, SLOT(
           pushButtonProximoClick()));
32     connect(ui.pushButtonFim, SIGNAL(clicked()), this, SLOT(
           fimClickPgTipificarAtributos()));
33     connect(ui.pushButtonAnterior, SIGNAL(clicked()), this, SLOT(
           pushButtonAnteriorClick()));
34
35     setupPgArquivoDelimitador();
36     ui.stackedWidgetInformacoes->setCurrentIndex(PAGE_ARQUIVO_DELIMITADOR);
37
38     ui.pushButtonAnterior->setVisible(false);
39     ui.pushButtonFim->setVisible(false);
40
41     ui.lineEditOutroDelimitador->setVisible(false);
42
43     setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
44 }
45
46 DialogImportacaoDados::~DialogImportacaoDados()
47 {
48     if (_progress)
49     {
50         _progress->close();
51         delete _progress;
52         _progress = NULL;
53     }
```

```
54
55      const void * address = static_cast<const void*>(&(*this));
56      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
57  }
58
59  void DialogImportacaoDados::pushButtonProximoClick()
60  {
61      setCursor(Qt::WaitCursor);
62
63      const int paginaAtual = ui.stackedWidgetInformacoes->currentIndex();
64
65      switch (paginaAtual)
66      {
67      case PAGE_ARQUIVO_DELIMITADOR:
68          {
69              proxClickPgArquivoDelimitador();
70          }
71          break;
72      case PAGE_SELECIONAR_ATRIBUTOS:
73          {
74              proxClickPgSelecionarAtributos();
75          }
76          break;
77      case PAGE_TIPIFICAR_ATRIBUTOS:
78          {
79              proxClickPgTipificarAtributos();
80          }
81          break;
82      case PAGE_INTERV_CAR_VALIDOS:
83          {
84              proxClickPgDefInterCatValidos();
85          }
86          break;
87      case PAGE_TRATAR_VALORES_INVALIDOS:
88          {
89              proxClickPgTratarForaIntervCat();
90          }
91          break;
92      case PAGE_SEL_TEC_OUTLIERS:
93          {
94              proxClickPgSelTecOutliers();
95          }
96          break;
97      case PAGE_TRATAR_OUTLIERS:
98          {
99              proxClickPgTratarOutliers();
100         }
101         break;
102     case PAGE_TRATAR_MISSINGS_INVALIDOS:
103         {
104             proxClickPgTratarMissingsInvalidos();
105         }
106         break;
107     }
108
109     setCursor(Qt::ArrowCursor);
110 }
111 void DialogImportacaoDados::pushButtonAnteriorClick()
112 {
113     setCursor(Qt::WaitCursor);
114
115     const int paginaAtual = ui.stackedWidgetInformacoes->currentIndex();
116
117     switch (paginaAtual)
118     {
119     case PAGE_SELECIONAR_ATRIBUTOS:
120         {
121             voltarClickPgSelecionarAtributos();
122         }
123         break;
124     case PAGE_TIPIFICAR_ATRIBUTOS:
125         {
126             voltarClickPgTipificarAtributos();
127         }
128         break;
129     case PAGE_INTERV_CAR_VALIDOS:
130         {
131             voltarClickPgDefInterCatValidos();
132         }
133         break;
134     case PAGE_TRATAR_VALORES_INVALIDOS:
135         {
```

```
136                 voltarClickPgTratarForaIntervCat();
137             }
138             break;
139         case PAGE_SEL_TEC_OUTLIERS:
140             {
141                 voltarClickPgSelTecOutliers();
142             }
143             break;
144         case PAGE_TRATAR_OUTLIERS:
145             {
146                 voltarClickPgTratarOutliers();
147             }
148             break;
149         case PAGE_TRATAR_MISSINGS_INVALIDOS:
150             {
151                 voltarClickPgTratarMissingsInvalidos();
152             }
153             break;
154         }
155 
156         setCursor(Qt::ArrowCursor);
157 }
158 void DialogImportacaoDados::fimColetaEstatisticas(const vector<string> *
        atributos)
159 {
160     if (!_fim)
161     {
162         setupPgDefInterCatValidos();
163         ui.stackedWidgetInformacoes->setCurrentIndex(
                PAGE_INTERV_CAR_VALIDOS);
164     }
165 
166     _progress->deleteLater();
167     _progress->close();
168 
169     delete atributos;
170 
171     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
172     ui.labelStatus->repaint();
173 
174     if (_fim)
175     {
176         accept();
177     }
178 }
179 
180 void DialogImportacaoDados::setupProgressBar()
181 {
182     _progress = new QProgressDialog("", "", 0, 100);
183     _progress->setWindowTitle(QString::fromLatin1("Aguarde"));
184     _progress->setWindowModality(Qt::ApplicationModal);
185     _progress->setWindowFlags(_progress->windowFlags() & ~Qt::
            WindowContextHelpButtonHint & ~Qt::WindowCloseButtonHint | Qt::
            MSWindowsFixedSizeDialogHint);
186     _progress->setCancelButton(NULL);
187     _progress->setValue(0);
188 }
189 
190 void DialogImportacaoDados::onUpdateProgresso(int *porcentagem)
191 {
192     _progress->setValue(min(*porcentagem, 99)); //evitar chegar 100 pois
            caso chegue ela se auto destroi
193     QCoreApplication::processEvents(QEventLoop::ExcludeUserInputEvents);
194 
195     delete porcentagem;
196 }
197 
198 void DialogImportacaoDados::setDados(vector<string> &atributos, const
        vector<TipoDado_AT> &tipos, const vector<vector<ValorBase *const>> &
        valoresColunas)
199 {
200     _atributos = atributos;
201     _tipos = tipos;
202     _valoresColunas = valoresColunas;
203     _fim = false;
204     iniciarColetarEstatisticas();
205 }
206 
207 void DialogImportacaoDados::iniciarColetarEstatisticas()
208 {
209     setupProgressBar();
210     _progress->setLabelText(QString::fromLatin1("Coletando estatísticas...")
```

```
        ));
211     _progress->show();
212     QCoreApplication::processEvents(QEventLoop::ExcludeUserInputEvents);
213
214     ThreadWorkerColetarEstatisticas *workerColetarEstatisticas = new
            ThreadWorkerColetarEstatisticas(_atributos, _tipos,
            _valoresColunas, this); //deletado quando destroi
            _threadController ou quando _threadController seta um novo worker
215     _threadController.setarWorker(workerColetarEstatisticas);
216     _threadController.iniciarExecucaoWorker(this);
217 }
```

Listagem B.33: dialog_importacao_dados.cpp

```
 1 #ifndef DIALOG_IMPORTACAO_DADOS_H
 2 #define DIALOG_IMPORTACAO_DADOS_H
 3
 4 #include "ui_dialog_importacao_dados.h"
 5
 6 #include <QFileDialog>
 7 #include <QMessageBox>
 8 #include <QProgressDialog>
 9
10 #include "widget_categorias_qual.h"
11 #include "widget_intervalos_quant.h"
12 #include "widget_tabela_editora.h"
13 #include "widget_param_diag_caixa.h"
14 #include "widget_param_zscore.h"
15
16 #include "thread_controller_dialog.h"
17 #include "dialog_progresso.h"
18
19 #define PAGE_ARQUIVO_DELIMITADOR 0
20 #define PAGE_SELECIONAR_ATRIBUTOS 1
21 #define PAGE_TIPIFICAR_ATRIBUTOS 2
22 #define PAGE_INTERV_CAR_VALIDOS 3
23 #define PAGE_TRATAR_VALORES_INVALIDOS 4
24 #define PAGE_SEL_TEC_OUTLIERS 5
25 #define PAGE_TRATAR_OUTLIERS 6
26 #define PAGE_TRATAR_MISSINGS_INVALIDOS 7
27
28 #define COLUNA_NOME_ATRIBUTO 0
29 #define COLUNA_TIPO_ATRIBUTO 1
30
31 using namespace std;
32
33 class DialogImportacaoDados : public DialogProgresso
34 {
35     Q_OBJECT
36
37 public:
38     DialogImportacaoDados(QWidget *parent);
39     ~DialogImportacaoDados();
40
41     void setDados(vector<string> &atributos, const vector<TipoDado_AT> &
            tipos, const vector<vector<ValorBase *const>> &valoresColunas);
42
43 private:
44     Ui::DialogImportacaoDados ui;
45
46     QProgressDialog *_progress;
47     ThreadControllerDialog _threadController;
48
49     void iniciarColetarEstatisticas();
50
51     //PG 00 e 01 arquivo e selecionar atributos
52     vector<vector<string>> _dadosColunasStr;
53     vector<string> _atributos;
54     string _diretorioArquivo;
55     string _delimitador;
56
57     bool verificarEntradas(string &erroMsg) const;
58     char delimitadorSelecionado() const;
59     bool carregarArquivo();
60     bool temAtributoSelecionado() const;
61     void filtrarAtributosNaoSelecionados();
62     //
```

```
63
64      //PG 02 tipificar
        ------------------------------------------------------------------
65      vector<TipoDado_AT> _tipos;
66      bool _fim;
67      vector<vector<ValorBase *const>> _valoresColunas;
68
69      void atualizaTiposAtributos();
70      void converterSetDados();
71      void coletarEstatisticas(const vector<vector<ValorBase *const>> &
            valoresColunas);
72      //
        ------------------------------------------------------------------

73
74      //PG 03 definir intervalos/categorias
        ------------------------------------------------------
75      vector<WidgetCategoriasQual*> _widgetsCategoriasQual;
76      vector<WidgetIntervalosQuant*> _widgetsIntervalosQuant;
77      map<string, vector<size_t>> _regsDescPorAtrib;
78      //
        ------------------------------------------------------------------

79
80      //PG 04 tratar valores fora do intervalo/categoria
        ---------------------------------------
81      WidgetTabelaEditora *_widgetTabelaEditoraTratarInterCateg;
82      //
        ------------------------------------------------------------------

83
84      //PG 05 tecnicas de outlier
        ------------------------------------------------------------
85      vector<WidgetParamDiagCaixa*> _widgetsParamDiagCaixa;
86      vector<WidgetParamZscore*> _widgetsParamZscore;
87      map<string, vector<size_t>> _indicesOutliersPorAtrib;
88      //
        ------------------------------------------------------------------

89
90      //PG 06 tratar outliers
        ------------------------------------------------------------
91      WidgetTabelaEditora *_widgetTabelaEditoraTratarOutliers;
92      //
        ------------------------------------------------------------------

93
94      //PG 06 tratar missings
        ------------------------------------------------------------
95      WidgetTabelaEditora *_widgetTabelaEditoraTratarMissingsInvalidos;
96      //
        ------------------------------------------------------------------

97
98      void setupPgArquivoDelimitador();
99      void setupPgSelecionarAtributos() const;
100     void setupPgTipificarAtributos();
101     void setupPgDefInterCatValidos();
102     void setupPgTratarForaIntervCat(const vector<size_t> &todosIndicesFora)
            ;
103     void setupPgSelTecOutliers();
104     void setupPgTratarOutliers(const vector<size_t> &todosIndicesOutliers);
105     void setupPgTratarMissingsInvalidos();
106
107     void proxClickPgArquivoDelimitador();
108     void proxClickPgSelecionarAtributos();
109     void proxClickPgTipificarAtributos();
110     void proxClickPgDefInterCatValidos();
111     void proxClickPgTratarForaIntervCat();
112     void proxClickPgSelTecOutliers();
113     void proxClickPgTratarOutliers();
114     void proxClickPgTratarMissingsInvalidos();
115
116     void voltarClickPgArquivoDelimitador();
117     void voltarClickPgSelecionarAtributos();
118     void voltarClickPgTipificarAtributos();
119     void voltarClickPgDefInterCatValidos();
120     void voltarClickPgTratarForaIntervCat();
121     void voltarClickPgSelTecOutliers();
122     void voltarClickPgTratarOutliers();
123     void voltarClickPgTratarMissingsInvalidos();
124
```

```
125     void setupProgressBar();
126
127 private slots:
128     void pushButtonProximoClick();
129     void pushButtonAnteriorClick();
130
131     void pushButtonAbrirArquivoClick() const;
132     void radioButtonClick() const;
133     void checkBoxSelecionarTodosClick() const;
134     void pushButtonGerarRegistrosMissings();
135     void fimClickPgTipificarAtributos();
136
137     void removerForaClick();
138     void removerOutlierClick();
139     void removerMissInvClick();
140
141 public slots:
142     void fimColetaEstatisticas(const vector<string> *atributos);
143 #ifndef TCC_RODOLFO
144     virtual void onUpdateProgresso(StatusThread*) {};
145 #endif
146     virtual void onUpdateProgresso(int *porcentagem);
147     void aplicarParamsDCTodos(const double fatorWI, const double fatorWS);
148     void aplicarParamsZSTodos(const double zMin, const double zMax);
149 };
150
151 #endif
```

Listagem B.34: dialog_importacao_dados.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogImportacaoDados</class>
4   <widget class="QDialog" name="DialogImportacaoDados">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>700</width>
10     <height>512</height>
11    </rect>
12   </property>
13   <property name="minimumSize">
14    <size>
15     <width>700</width>
16     <height>512</height>
17    </size>
18   </property>
19   <property name="windowTitle">
20    <string>Importar Dados</string>
21   </property>
22   <layout class="QVBoxLayout" name="verticalLayout_12">
23    <item>
24     <widget class="QLabel" name="labelTitulo">
25      <property name="sizePolicy">
26       <sizepolicy hsizetype="Expanding" vsizetype="Fixed">
27        <horstretch>0</horstretch>
28        <verstretch>0</verstretch>
29       </sizepolicy>
30      </property>
31      <property name="font">
32       <font>
33        <pointsize>11</pointsize>
34        <weight>75</weight>
35        <bold>true</bold>
36       </font>
37      </property>
38      <property name="text">
39       <string>Análise e Tratamento de Dados</string>
40      </property>
41     </widget>
42    </item>
43    <item>
44     <widget class="QStackedWidget" name="stackedWidgetInformacoes">
45      <property name="sizePolicy">
46       <sizepolicy hsizetype="Preferred" vsizetype="Minimum">
47        <horstretch>0</horstretch>
48        <verstretch>0</verstretch>
49       </sizepolicy>
```

```
50        </property>
51        <property name="currentIndex">
52         <number>0</number>
53        </property>
54        <widget class="QWidget" name="pageArquivoDelimitador">
55         <layout class="QVBoxLayout" name="verticalLayout_11">
56          <property name="leftMargin">
57           <number>0</number>
58          </property>
59          <property name="topMargin">
60           <number>0</number>
61          </property>
62          <property name="rightMargin">
63           <number>0</number>
64          </property>
65          <property name="bottomMargin">
66           <number>0</number>
67          </property>
68          <item>
69           <layout class="QVBoxLayout" name="verticalLayout_5">
70            <item>
71             <widget class="QGroupBox" name="groupBoxArquivo">
72              <property name="sizePolicy">
73               <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
74                <horstretch>0</horstretch>
75                <verstretch>0</verstretch>
76               </sizepolicy>
77              </property>
78              <property name="title">
79               <string>Local</string>
80              </property>
81              <layout class="QVBoxLayout" name="verticalLayout_3">
82               <item>
83                <layout class="QHBoxLayout" name="horizontalLayout_2">
84                 <item>
85                  <widget class="QLineEdit" name="lineEditArquivo">
86                   <property name="text">
87                    <string/>
88                   </property>
89                   <property name="readOnly">
90                    <bool>true</bool>
91                   </property>
92                  </widget>
93                 </item>
94                 <item>
95                  <widget class="QPushButton" name="pushButtonAbrirArquivo">
96                   <property name="text">
97                    <string>Abrir</string>
98                   </property>
99                  </widget>
100                </item>
101               </layout>
102              </item>
103             </layout>
104            </widget>
105           </item>
106           <item>
107            <widget class="QGroupBox" name="groupBox">
108             <property name="sizePolicy">
109              <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
110               <horstretch>0</horstretch>
111               <verstretch>0</verstretch>
112              </sizepolicy>
113             </property>
114             <property name="title">
115              <string>Delimitador</string>
116             </property>
117             <layout class="QVBoxLayout" name="verticalLayout_4">
118              <item>
119               <layout class="QGridLayout" name="gridLayout">
120                <item row="0" column="0">
121                 <widget class="QRadioButton" name="radioButtonPontoVirgula">
122                  <property name="text">
123                   <string>Ponto e vírgula</string>
124                  </property>
125                  <property name="checked">
126                   <bool>true</bool>
127                  </property>
128                 </widget>
129                </item>
130                <item row="0" column="1">
131                 <spacer name="horizontalSpacer_2">
```

```
132                  <property name="orientation">
133                   <enum>Qt::Horizontal</enum>
134                  </property>
135                  <property name="sizeHint" stdset="0">
136                   <size>
137                    <width>40</width>
138                    <height>20</height>
139                   </size>
140                  </property>
141                 </spacer>
142                </item>
143                <item row="0" column="2">
144                 <widget class="QRadioButton" name="radioButtonTabulacao">
145                  <property name="text">
146                   <string>Tabulação</string>
147                  </property>
148                 </widget>
149                </item>
150                <item row="1" column="0">
151                 <widget class="QRadioButton" name="radioButtonEspaco">
152                  <property name="text">
153                   <string>Espaço</string>
154                  </property>
155                 </widget>
156                </item>
157                <item row="1" column="1">
158                 <spacer name="horizontalSpacer_3">
159                  <property name="orientation">
160                   <enum>Qt::Horizontal</enum>
161                  </property>
162                  <property name="sizeHint" stdset="0">
163                   <size>
164                    <width>40</width>
165                    <height>20</height>
166                   </size>
167                  </property>
168                 </spacer>
169                </item>
170                <item row="1" column="2">
171                 <layout class="QHBoxLayout" name="horizontalLayout_3">
172                  <item>
173                   <widget class="QRadioButton" name="radioButtonOutro">
174                    <property name="text">
175                     <string>Outro:</string>
176                    </property>
177                   </widget>
178                  </item>
179                  <item>
180                   <widget class="QLineEdit" name="lineEditOutroDelimitador">
181                    <property name="sizePolicy">
182                     <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
183                      <horstretch>0</horstretch>
184                      <verstretch>0</verstretch>
185                     </sizepolicy>
186                    </property>
187                    <property name="maxLength">
188                     <number>1</number>
189                    </property>
190                    <property name="readOnly">
191                     <bool>false</bool>
192                    </property>
193                   </widget>
194                  </item>
195                 </layout>
196                </item>
197               </layout>
198              </item>
199             </layout>
200            </widget>
201           </item>
202           <item>
203            <spacer name="verticalSpacer">
204             <property name="orientation">
205              <enum>Qt::Vertical</enum>
206             </property>
207             <property name="sizeHint" stdset="0">
208              <size>
209               <width>20</width>
210               <height>40</height>
211              </size>
212             </property>
213            </spacer>
```

```
214             </item>
215             <item>
216              <widget class="QCheckBox" name="checkBoxPrimeiraLinhaNome">
217               <property name="text">
218                <string>A primeira linha possui o nome dos atributos</string>
219               </property>
220               <property name="checked">
221                <bool>true</bool>
222               </property>
223              </widget>
224             </item>
225            </layout>
226           </item>
227          </layout>
228         </widget>
229         <widget class="QWidget" name="pageSelecionarAtributos">
230          <layout class="QVBoxLayout" name="verticalLayout_10">
231           <property name="leftMargin">
232            <number>0</number>
233           </property>
234           <property name="topMargin">
235            <number>0</number>
236           </property>
237           <property name="rightMargin">
238            <number>0</number>
239           </property>
240           <property name="bottomMargin">
241            <number>0</number>
242           </property>
243           <item>
244            <widget class="QGroupBox" name="groupBoxSelecionarAtributos">
245             <property name="title">
246              <string>Selecione os atributos</string>
247             </property>
248             <layout class="QVBoxLayout" name="verticalLayout_7">
249              <item>
250               <layout class="QVBoxLayout" name="verticalLayout_6">
251                <item>
252                 <widget class="QListWidget" name="listWidgetNomeAtributos"/>
253                </item>
254                <item>
255                 <widget class="QCheckBox" name="checkBoxSelecionarTodos">
256                  <property name="text">
257                   <string>Selecionar todos.</string>
258                  </property>
259                  <property name="checked">
260                   <bool>true</bool>
261                  </property>
262                 </widget>
263                </item>
264               </layout>
265              </item>
266             </layout>
267            </widget>
268           </item>
269          </layout>
270         </widget>
271         <widget class="QWidget" name="pageTipificarAtributos">
272          <layout class="QVBoxLayout" name="verticalLayout_9">
273           <property name="leftMargin">
274            <number>0</number>
275           </property>
276           <property name="topMargin">
277            <number>0</number>
278           </property>
279           <property name="rightMargin">
280            <number>0</number>
281           </property>
282           <property name="bottomMargin">
283            <number>0</number>
284           </property>
285           <item>
286            <widget class="QGroupBox" name="groupBoxEditarTiposAtributos">
287             <property name="title">
288              <string>Escolha os tipos dos atributos</string>
289             </property>
290             <layout class="QVBoxLayout" name="verticalLayout_8">
291              <item>
292               <widget class="QTableWidget" name="
                      tableWidgetEditarTiposAtributos">
293                <column>
294                 <property name="text">
```

```
295                 <string>Atributo</string>
296                </property>
297               </column>
298               <column>
299                <property name="text">
300                 <string>Tipo</string>
301                </property>
302               </column>
303              </widget>
304             </item>
305            </layout>
306           </widget>
307          </item>
308         </layout>
309        </widget>
310        <widget class="QWidget" name="pageDefinirInterCat">
311         <layout class="QVBoxLayout" name="verticalLayout_14">
312          <property name="leftMargin">
313           <number>0</number>
314          </property>
315          <property name="topMargin">
316           <number>0</number>
317          </property>
318          <property name="rightMargin">
319           <number>0</number>
320          </property>
321          <property name="bottomMargin">
322           <number>0</number>
323          </property>
324          <item>
325           <widget class="QGroupBox" name="groupBox_5">
326            <property name="title">
327             <string>Definir intervalos/categorias válidos</string>
328            </property>
329            <layout class="QVBoxLayout" name="verticalLayout_22">
330             <item>
331              <widget class="QTabWidget" name="tabWidgetValoresValidos"/>
332             </item>
333            </layout>
334           </widget>
335          </item>
336         </layout>
337        </widget>
338        <widget class="QWidget" name="pageTratarForaInterCat">
339         <layout class="QVBoxLayout" name="verticalLayout_13">
340          <property name="leftMargin">
341           <number>0</number>
342          </property>
343          <property name="topMargin">
344           <number>0</number>
345          </property>
346          <property name="rightMargin">
347           <number>0</number>
348          </property>
349          <property name="bottomMargin">
350           <number>0</number>
351          </property>
352          <item>
353           <widget class="QGroupBox" name="groupBox_4">
354            <property name="title">
355             <string>Tratar valores fora do intervalo/categoria</string>
356            </property>
357            <layout class="QVBoxLayout" name="verticalLayout_2">
358             <item>
359              <layout class="QHBoxLayout" name="horizontalLayout_14">
360               <item>
361                <widget class="QGroupBox" name="groupBox_11">
362                 <property name="title">
363                  <string>Remover Destacados</string>
364                 </property>
365                 <layout class="QHBoxLayout" name="horizontalLayout_13">
366                  <item>
367                   <widget class="QComboBox" name="comboBoxRemoverFora"/>
368                  </item>
369                  <item>
370                   <widget class="QPushButton" name="pushButtonRemoverFora">
371                    <property name="text">
372                     <string>Remover</string>
373                    </property>
374                   </widget>
375                  </item>
376                 </layout>
```

```
377                </widget>
378               </item>
379               <item>
380                <spacer name="horizontalSpacer_7">
381                 <property name="orientation">
382                  <enum>Qt::Horizontal</enum>
383                 </property>
384                 <property name="sizeHint" stdset="0">
385                  <size>
386                   <width>40</width>
387                   <height>20</height>
388                  </size>
389                 </property>
390                </spacer>
391               </item>
392              </layout>
393             </item>
394             <item>
395              <widget class="QWidget" name="widgetPgTratarForaInterCateg"
                       native="true">
396               <property name="sizePolicy">
397                <sizepolicy hsizetype="Expanding" vsizetype="Expanding">
398                 <horstretch>0</horstretch>
399                 <verstretch>0</verstretch>
400                </sizepolicy>
401               </property>
402               <layout class="QVBoxLayout" name="verticalLayout_16">
403                <property name="leftMargin">
404                 <number>0</number>
405                </property>
406                <property name="topMargin">
407                 <number>0</number>
408                </property>
409                <property name="rightMargin">
410                 <number>0</number>
411                </property>
412                <property name="bottomMargin">
413                 <number>0</number>
414                </property>
415               </layout>
416              </widget>
417             </item>
418            </layout>
419           </widget>
420          </item>
421         </layout>
422        </widget>
423        <widget class="QWidget" name="pageTecnicasOutlier">
424         <layout class="QVBoxLayout" name="verticalLayout_15">
425          <property name="leftMargin">
426           <number>0</number>
427          </property>
428          <property name="topMargin">
429           <number>0</number>
430          </property>
431          <property name="rightMargin">
432           <number>0</number>
433          </property>
434          <property name="bottomMargin">
435           <number>0</number>
436          </property>
437          <item>
438           <widget class="QGroupBox" name="groupBox_2">
439            <property name="title">
440             <string>Detecção de outliers</string>
441            </property>
442            <layout class="QVBoxLayout" name="verticalLayout_19">
443             <item>
444              <layout class="QHBoxLayout" name="horizontalLayout_5">
445               <item>
446                <layout class="QVBoxLayout" name="verticalLayout_17">
447                 <item>
448                  <widget class="QCheckBox" name="checkBoxDiagramaCaixa">
449                   <property name="minimumSize">
450                    <size>
451                     <width>115</width>
452                     <height>0</height>
453                    </size>
454                   </property>
455                   <property name="text">
456                    <string>Diagrama de Caixa</string>
457                   </property>
```

```
458                 </widget>
459                </item>
460                <item>
461                 <spacer name="verticalSpacer_2">
462                  <property name="orientation">
463                   <enum>Qt::Vertical</enum>
464                  </property>
465                  <property name="sizeHint" stdset="0">
466                   <size>
467                    <width>20</width>
468                    <height>40</height>
469                   </size>
470                  </property>
471                 </spacer>
472                </item>
473               </layout>
474              </item>
475              <item>
476               <widget class="QTabWidget" name="tabWidgetParamsDiagCaixa"/>
477              </item>
478             </layout>
479            </item>
480            <item>
481             <layout class="QHBoxLayout" name="horizontalLayout_6">
482              <item>
483               <layout class="QVBoxLayout" name="verticalLayout_18">
484                <item>
485                 <widget class="QCheckBox" name="checkBoxZScore">
486                  <property name="minimumSize">
487                   <size>
488                    <width>115</width>
489                    <height>0</height>
490                   </size>
491                  </property>
492                  <property name="text">
493                   <string>Z-Score</string>
494                  </property>
495                 </widget>
496                </item>
497                <item>
498                 <spacer name="verticalSpacer_3">
499                  <property name="orientation">
500                   <enum>Qt::Vertical</enum>
501                  </property>
502                  <property name="sizeHint" stdset="0">
503                   <size>
504                    <width>20</width>
505                    <height>40</height>
506                   </size>
507                  </property>
508                 </spacer>
509                </item>
510               </layout>
511              </item>
512              <item>
513               <widget class="QTabWidget" name="tabWidgetParamsZScore"/>
514              </item>
515             </layout>
516            </item>
517           </layout>
518          </widget>
519         </item>
520        </layout>
521       </widget>
522       <widget class="QWidget" name="pageTratarOutliers">
523        <layout class="QVBoxLayout" name="verticalLayout_21">
524         <property name="leftMargin">
525          <number>0</number>
526         </property>
527         <property name="topMargin">
528          <number>0</number>
529         </property>
530         <property name="rightMargin">
531          <number>0</number>
532         </property>
533         <property name="bottomMargin">
534          <number>0</number>
535         </property>
536         <item>
537          <widget class="QGroupBox" name="groupBox_3">
538           <property name="title">
539            <string>Tratar outliers</string>
```

```
540             </property>
541             <layout class="QVBoxLayout" name="verticalLayout">
542              <item>
543               <layout class="QHBoxLayout" name="horizontalLayout_7">
544                <item>
545                 <widget class="QGroupBox" name="groupBox_7">
546                  <property name="title">
547                   <string>Remover Destacados</string>
548                  </property>
549                  <layout class="QHBoxLayout" name="horizontalLayout_8">
550                   <item>
551                    <widget class="QComboBox" name="comboBoxRemoverOutlier"/>
552                   </item>
553                   <item>
554                    <widget class="QPushButton" name="pushButtonRemoverOutlier"
                          >
555                     <property name="text">
556                      <string>Remover</string>
557                     </property>
558                    </widget>
559                   </item>
560                  </layout>
561                 </widget>
562                </item>
563                <item>
564                 <widget class="QGroupBox" name="groupBox_8">
565                  <property name="title">
566                   <string>Substituir Destacados</string>
567                  </property>
568                  <layout class="QHBoxLayout" name="horizontalLayout_9">
569                   <item>
570                    <widget class="QPushButton" name="
                          pushButtonSubstituirOutlier">
571                     <property name="text">
572                      <string>Substituir</string>
573                     </property>
574                    </widget>
575                   </item>
576                  </layout>
577                 </widget>
578                </item>
579                <item>
580                 <spacer name="horizontalSpacer_5">
581                  <property name="orientation">
582                   <enum>Qt::Horizontal</enum>
583                  </property>
584                  <property name="sizeHint" stdset="0">
585                   <size>
586                    <width>40</width>
587                    <height>20</height>
588                   </size>
589                  </property>
590                 </spacer>
591                </item>
592               </layout>
593              </item>
594              <item>
595               <widget class="QWidget" name="widgetPgTratarOutlier" native="
                     true">
596                <property name="sizePolicy">
597                 <sizepolicy hsizetype="Expanding" vsizetype="Expanding">
598                  <horstretch>0</horstretch>
599                  <verstretch>0</verstretch>
600                 </sizepolicy>
601                </property>
602                <layout class="QVBoxLayout" name="verticalLayout_20">
603                 <property name="leftMargin">
604                  <number>0</number>
605                 </property>
606                 <property name="topMargin">
607                  <number>0</number>
608                 </property>
609                 <property name="rightMargin">
610                  <number>0</number>
611                 </property>
612                 <property name="bottomMargin">
613                  <number>0</number>
614                 </property>
615                </layout>
616               </widget>
617              </item>
618             </layout>
```

```
619           </widget>
620          </item>
621         </layout>
622        </widget>
623        <widget class="QWidget" name="pageTratarMissingsInvalidos">
624         <layout class="QVBoxLayout" name="verticalLayout_23">
625          <item>
626           <widget class="QGroupBox" name="groupBox_6">
627            <property name="title">
628             <string>Tratar missings ou inválidos</string>
629            </property>
630            <layout class="QVBoxLayout" name="verticalLayout_25">
631             <item>
632              <layout class="QHBoxLayout" name="horizontalLayout_4">
633               <item>
634                <widget class="QCheckBox" name="checkBoxUtilizarVarRef">
635                 <property name="text">
636                  <string>Utilizar variável de referência</string>
637                 </property>
638                </widget>
639               </item>
640               <item>
641                <widget class="QComboBox" name="comboBoxVariaveisReferencia">
642                 <property name="enabled">
643                  <bool>false</bool>
644                 </property>
645                </widget>
646               </item>
647               <item>
648                <widget class="QPushButton" name="
                        pushButtonGerarRegistrosMissings">
649                 <property name="enabled">
650                  <bool>false</bool>
651                 </property>
652                 <property name="text">
653                  <string>Gerar registros faltantes</string>
654                 </property>
655                </widget>
656               </item>
657               <item>
658                <spacer name="horizontalSpacer_4">
659                 <property name="orientation">
660                  <enum>Qt::Horizontal</enum>
661                 </property>
662                 <property name="sizeHint" stdset="0">
663                  <size>
664                   <width>40</width>
665                   <height>20</height>
666                  </size>
667                 </property>
668                </spacer>
669               </item>
670              </layout>
671             </item>
672             <item>
673              <layout class="QHBoxLayout" name="horizontalLayout_10">
674               <item>
675                <widget class="QGroupBox" name="groupBox_9">
676                 <property name="title">
677                  <string>Remover Destacados</string>
678                 </property>
679                 <layout class="QHBoxLayout" name="horizontalLayout_11">
680                  <item>
681                   <widget class="QComboBox" name="comboBoxRemoverMissInv"/>
682                  </item>
683                  <item>
684                   <widget class="QPushButton" name="pushButtonRemoverMissInv"
                         >
685                    <property name="text">
686                     <string>Remover</string>
687                    </property>
688                   </widget>
689                  </item>
690                 </layout>
691                </widget>
692               </item>
693               <item>
694                <widget class="QGroupBox" name="groupBox_10">
695                 <property name="title">
696                  <string>Substituir Destacados</string>
697                 </property>
698                 <layout class="QHBoxLayout" name="horizontalLayout_12">
```

```
699                  <item>
700                   <widget class="QPushButton" name="
                            pushButtonSubstituirMissInv">
701                    <property name="text">
702                     <string>Substituir</string>
703                    </property>
704                   </widget>
705                  </item>
706                 </layout>
707                </widget>
708               </item>
709               <item>
710                <spacer name="horizontalSpacer_6">
711                 <property name="orientation">
712                  <enum>Qt::Horizontal</enum>
713                 </property>
714                 <property name="sizeHint" stdset="0">
715                  <size>
716                   <width>40</width>
717                   <height>20</height>
718                  </size>
719                 </property>
720                </spacer>
721               </item>
722              </layout>
723             </item>
724             <item>
725              <widget class="QWidget" name="widgetPgTratarMissingsInvalidos"
                       native="true">
726               <property name="sizePolicy">
727                <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
728                 <horstretch>0</horstretch>
729                 <verstretch>0</verstretch>
730                </sizepolicy>
731               </property>
732               <layout class="QVBoxLayout" name="verticalLayout_24">
733                <property name="leftMargin">
734                 <number>0</number>
735                </property>
736                <property name="topMargin">
737                 <number>0</number>
738                </property>
739                <property name="rightMargin">
740                 <number>0</number>
741                </property>
742                <property name="bottomMargin">
743                 <number>0</number>
744                </property>
745               </layout>
746              </widget>
747             </item>
748            </layout>
749           </widget>
750          </item>
751         </layout>
752        </widget>
753       </widget>
754      </item>
755      <item>
756       <layout class="QHBoxLayout" name="horizontalLayout">
757        <item>
758         <widget class="QLabel" name="labelStatus">
759          <property name="minimumSize">
760           <size>
761            <width>200</width>
762            <height>0</height>
763           </size>
764          </property>
765          <property name="text">
766           <string>Pronto.</string>
767          </property>
768         </widget>
769        </item>
770        <item>
771         <spacer name="horizontalSpacer">
772          <property name="orientation">
773           <enum>Qt::Horizontal</enum>
774          </property>
775          <property name="sizeHint" stdset="0">
776           <size>
777            <width>40</width>
778            <height>20</height>
```

```
779           </size>
780          </property>
781         </spacer>
782        </item>
783        <item>
784         <widget class="QPushButton" name="pushButtonAnterior">
785          <property name="text">
786           <string>Anterior</string>
787          </property>
788         </widget>
789        </item>
790        <item>
791         <widget class="QPushButton" name="pushButtonFim">
792          <property name="text">
793           <string>Fim</string>
794          </property>
795         </widget>
796        </item>
797        <item>
798         <widget class="QPushButton" name="pushButtonProximo">
799          <property name="text">
800           <string>Próximo</string>
801          </property>
802         </widget>
803        </item>
804       </layout>
805      </item>
806     </layout>
807    </widget>
808    <layoutdefault spacing="6" margin="11"/>
809    <resources/>
810    <connections/>
811   </ui>
```

Listagem B.35: dialog_importacao_dados.ui

```
1  #include "dialog_importacao_dados.h"
2
3  #include "utilidades.h"
4
5  void DialogImportacaoDados::proxClickPgArquivoDelimitador()
6  {
7      string erroMsg;
8      bool entradasValidas = verificarEntradas(erroMsg);
9
10     if (entradasValidas)
11     {
12         _diretorioArquivo = ui.lineEditArquivo->text().toLatin1().data();
13         _delimitador = delimitadorSelecionado();
14
15         if (carregarArquivo())
16         {
17             setupPgSelecionarAtributos();
18             ui.pushButtonAnterior->setVisible(true);
19             ui.stackedWidgetInformacoes->setCurrentIndex(
                   PAGE_SELECIONAR_ATRIBUTOS);
20         }
21         else
22         {
23             QMessageBox::critical(NULL, QString::fromLatin1("Erro"),
                       QString::fromLatin1("Ocorreu um erro ao tentar carregar o
                       arquivo!"));
24         }
25     }
26     else
27     {
28         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1(erroMsg.c_str()));
29     }
30 }
31 void DialogImportacaoDados::voltarClickPgArquivoDelimitador()
32 {
33
34 }
35 void DialogImportacaoDados::setupPgArquivoDelimitador()
36 {
37     connect(ui.pushButtonAbrirArquivo, SIGNAL(clicked()), this, SLOT(
            pushButtonAbrirArquivoClick()));
38     connect(ui.radioButtonPontoVirgula, SIGNAL(clicked()), this, SLOT(
            radioButtonClick()));
```

```
39      connect(ui.radioButtonEspaco, SIGNAL(clicked()), this, SLOT(
            radioButtonClick()));
40      connect(ui.radioButtonTabulacao, SIGNAL(clicked()), this, SLOT(
            radioButtonClick()));
41      connect(ui.radioButtonOutro, SIGNAL(clicked()), this, SLOT(
            radioButtonClick()));
42  }
43  bool DialogImportacaoDados::verificarEntradas(string &erroMsg) const
44  {
45      bool entradasValidas = true;
46
47      if (ui.lineEditArquivo->text() == "")
48      {
49          erroMsg += "Nenhum arquivo Selecionado.\n";
50          entradasValidas = false;
51      }
52
53      if (ui.radioButtonOutro->isChecked() && ui.lineEditOutroDelimitador->
            text() == "")
54      {
55          erroMsg += "Nenhum delimitador inserido.";
56          entradasValidas = false;
57      }
58
59      return entradasValidas;
60  }
61  bool DialogImportacaoDados::carregarArquivo()
62  {
63      ui.labelStatus->setText(QString::fromLatin1("Carregando arquivo..."));
64      ui.labelStatus->repaint();
65
66      bool sucesso = true;
67
68      _dadosColunasStr.clear();
69      _atributos.clear();
70
71      vector<vector<string>> dadosEmLinhas = vector<vector<string>>();
72
73      setupProgressBar();
74      _progress->setLabelText(QString::fromLatin1("Carregando arquivo..."));
75      _progress->show();
76      QCoreApplication::processEvents(QEventLoop::ExcludeUserInputEvents);
77
78      //50%
79      if (sucesso && !Utilidades::parseCVSDados(_delimitador,
            _diretorioArquivo, dadosEmLinhas, _atributos, _progress))
80      {
81          sucesso = false;
82      }
83
84      const size_t totalColunas = _atributos.size();
85      const size_t totalLinhas = dadosEmLinhas.size();
86      for (size_t i = 0; i < totalColunas; i++)
87      {
88          _dadosColunasStr.push_back(vector<string>());
89          _dadosColunasStr.back().reserve(totalLinhas);
90      }
91
92      const int taxaNotificacao = totalLinhas <= 10 ? 10 : totalLinhas / 10;
93      size_t i = 0;
94
95      //50%
96      for (vector<vector<string>>::const_iterator citLinha = dadosEmLinhas.
            cbegin();
97          citLinha != dadosEmLinhas.cend();
98          citLinha++, i++)
99      {
100         size_t coluna = 0;
101         for (vector<string>::const_iterator citColuna = citLinha->cbegin();
102             citColuna != citLinha->cend();
103             citColuna++, coluna++)
104         {
105             _dadosColunasStr[coluna].push_back(*citColuna);
106         }
107
108         if ((i % taxaNotificacao) == 0)
109         {
110             _progress->setValue(min((int)(i * 1.0 / totalLinhas * 50 + 50),
                    99));
111             QCoreApplication::processEvents(QEventLoop::
                    ExcludeUserInputEvents);
112         }
```

```
113     }
114
115     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
116     ui.labelStatus->repaint();
117
118     _progress->deleteLater();
119     _progress->close();
120
121     return sucesso;
122 }
123 void DialogImportacaoDados::pushButtonAbrirArquivoClick() const
124 {
125     QString fileName = QFileDialog::getOpenFileName(NULL, QString::
            fromLatin1("Selecione o arquivo"), "", QString::fromLatin1("All
            files (*.*)"));
126     if (!fileName.isEmpty())
127     {
128         ui.lineEditArquivo->setText(fileName);
129     }
130 }
131 void DialogImportacaoDados::radioButtonClick() const
132 {
133     ui.lineEditOutroDelimitador->setVisible(ui.radioButtonOutro->isChecked
            ());
134 }
135 char DialogImportacaoDados::delimitadorSelecionado() const
136 {
137     if (ui.radioButtonPontoVirgula->isChecked())
138     {
139         return ';';
140     }
141     else if (ui.radioButtonEspaco->isChecked())
142     {
143         return ' ';
144     }
145     else if (ui.radioButtonTabulacao->isChecked())
146     {
147         return '\t';
148     }
149     else /*if (ui.radioButtonOutro->isChecked())*/
150     {
151         return ui.lineEditOutroDelimitador->text().toLatin1().data()[0];
152     }
153 }
```

Listagem B.36: dialog_importacao_dados_pg_00_arquivo.cpp

```
1  #include "dialog_importacao_dados.h"
2
3  void DialogImportacaoDados::proxClickPgSelecionarAtributos()
4  {
5      if (temAtributoSelecionado())
6      {
7          filtrarAtributosNaoSelecionados();
8          setupPgTipificarAtributos();
9          ui.stackedWidgetInformacoes->setCurrentIndex(
               PAGE_TIPIFICAR_ATRIBUTOS);
10     }
11     else
12     {
13         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
               fromLatin1("Não há atributos selecionados"));
14     }
15
16 }
17 void DialogImportacaoDados::voltarClickPgSelecionarAtributos()
18 {
19     ui.stackedWidgetInformacoes->setCurrentIndex(PAGE_ARQUIVO_DELIMITADOR);
20     ui.pushButtonAnterior->setVisible(false);
21 }
22 void DialogImportacaoDados::setupPgSelecionarAtributos() const
23 {
24     connect(ui.checkBoxSelecionarTodos, SIGNAL(clicked()), this, SLOT(
           checkBoxSelecionarTodosClick()));
25
26     ui.labelStatus->setText(QString::fromLatin1("Preenchendo a tabela com
           atributos..."));
27     ui.labelStatus->repaint();
28
```

```
29      ui.listWidgetNomeAtributos->clear();
30
31      QListWidgetItem *item;
32      for (vector<string>::const_iterator citAtributo = _atributos.cbegin();
33           citAtributo != _atributos.cend();
34           citAtributo++)
35      {
36          item = new QListWidgetItem();
37          item->setData(Qt::DisplayRole, QString::fromLatin1(citAtributo->
                c_str()));
38          item->setData(Qt::CheckStateRole, Qt::Checked);
39
40          ui.listWidgetNomeAtributos->addItem(item);
41      }
42      ui.labelStatus->setText(QString::fromLatin1("Pronto."));
43      ui.labelStatus->repaint();
44
45  }
46  bool DialogImportacaoDados::temAtributoSelecionado() const
47  {
48      QListWidgetItem *item;
49
50      const int count =  ui.listWidgetNomeAtributos->count();
51
52      for (int i = 0; i < count; i++)
53      {
54          item = ui.listWidgetNomeAtributos->item(i);
55          if (item->checkState() == Qt::Checked)
56          {
57              return true;
58          }
59      }
60
61      return false;
62  }
63  void DialogImportacaoDados::filtrarAtributosNaoSelecionados()
64  {
65      ui.labelStatus->setText(QString::fromLatin1("Selecionando os atributos
            ..."));
66      ui.labelStatus->repaint();
67
68      vector<string>::const_iterator citAtributo = _atributos.cbegin();
69      vector<vector<string>>::const_iterator citColuna = _dadosColunasStr.
            cbegin();
70
71      const int count = ui.listWidgetNomeAtributos->count();
72
73      for (int i = 0; i < count; i++)
74      {
75          if (ui.listWidgetNomeAtributos->item(i)->checkState() != Qt::
                Checked)
76          {
77              citAtributo = _atributos.erase(citAtributo);
78              citColuna = _dadosColunasStr.erase(citColuna);
79          }
80          else
81          {
82              ++citAtributo;
83              ++citColuna;
84          }
85      }
86
87      ui.labelStatus->setText(QString::fromLatin1("Pronto."));
88      ui.labelStatus->repaint();
89  }
90  void DialogImportacaoDados::checkBoxSelecionarTodosClick() const
91  {
92      Qt::CheckState state = ui.checkBoxSelecionarTodos->checkState();
93
94      const int count = ui.listWidgetNomeAtributos->count();
95
96      for (int i = 0; i < count; i++)
97      {
98          ui.listWidgetNomeAtributos->item(i)->setCheckState(state);
99      }
100 }
```

Listagem B.37: dialog_importacao_dados_pg_01_selecionar_atribs.cpp

```
1  #include "dialog_importacao_dados.h"
2
3  void DialogImportacaoDados::proxClickPgTipificarAtributos()
4  {
5      const int count = ui.tableWidgetEditarTiposAtributos->rowCount();
6
7      for (size_t i = 0; i < count; i++)
8      {
9          QComboBox *combo = (QComboBox*) ui.tableWidgetEditarTiposAtributos
               ->cellWidget(i, COLUNA_TIPO_ATRIBUTO);
10
11         const QString atual = combo->currentText();
12
13         if (atual == QString::fromLatin1(""))
14         {
15             QMessageBox::warning(NULL, QString::fromLatin1("Aviso"),
                   QString::fromLatin1("Alguns atributos não foram
                   tipificados!"));
16             return;
17         }
18     }
19
20     atualizaTiposAtributos();
21
22     ui.labelStatus->setText(QString::fromLatin1("Adequando dados..."));
23     ui.labelStatus->repaint();
24
25     converterSetDados();
26
27     ui.pushButtonFim->setVisible(false);
28
29     //emit dadosCarregados(_dados, _atributos, _tipos);
30     //accept();
31 }
32 void DialogImportacaoDados::voltarClickPgTipificarAtributos()
33 {
34     if (carregarArquivo())
35     {
36         setupPgSelecionarAtributos();
37         ui.stackedWidgetInformacoes->setCurrentIndex(
               PAGE_SELECIONAR_ATRIBUTOS);
38     }
39     else
40     {
41         QMessageBox::critical(NULL, QString::fromLatin1("Erro"), QString::
               fromLatin1("Ocorreu um erro ao tentar carregar o arquivo!"));
42     }
43 }
44 void DialogImportacaoDados::fimClickPgTipificarAtributos()
45 {
46     _fim = true;
47     proxClickPgTipificarAtributos();
48 }
49 void DialogImportacaoDados::setupPgTipificarAtributos()
50 {
51     _fim = false;
52
53     ui.labelStatus->setText(QString::fromLatin1("Preenchendo a lista com
           tipos dos atributos..."));
54     ui.labelStatus->repaint();
55
56     QStringList tiposList = QStringList();
57     tiposList.append(QString::fromLatin1(""));
58     tiposList.append(QString::fromLatin1("Qualitativo"));
59     tiposList.append(QString::fromLatin1("Quantitativo Contínuo"));
60     tiposList.append(QString::fromLatin1("Quantitativo Discreto"));
61
62     /*_tipos.clear();
63     for (size_t i = 0; i < _atributos.size(); i++)
64     {
65         _tipos.push_back(QUANT_CONTINUO); //default
66     }*/
67
68     const size_t count = _atributos.size();
69
70     ui.tableWidgetEditarTiposAtributos->setRowCount(count);
71
72     for (size_t i = 0; i < count; i++)
73     {
74         QTableWidgetItem *item = new QTableWidgetItem(QString::fromLatin1(
               _atributos[i].c_str()));
75
```

```
76          ui.tableWidgetEditarTiposAtributos->setItem(i, COLUNA_NOME_ATRIBUTO
                , item);
77
78          QComboBox *combo = new QComboBox(ui.tableWidgetEditarTiposAtributos
                );
79
80          combo->addItems(tiposList);
81
82          ui.tableWidgetEditarTiposAtributos->setCellWidget(i,
                COLUNA_TIPO_ATRIBUTO, combo);
83
84          if (i < _tipos.size())
85          {
86              switch (_tipos[i])
87              {
88              case QUALITATIVO:
89                  {
90                      combo->setCurrentIndex(combo->findText(QString::
                            fromLatin1("Qualitativo")));
91                  }
92                  break;
93
94              case QUANT_CONTINUO:
95                  {
96                      combo->setCurrentIndex(combo->findText(QString::
                            fromLatin1("Quantitativo Contínuo")));
97                  }
98                  break;
99              case QUANT_DISCRETO:
100                 {
101                     combo->setCurrentIndex(combo->findText(QString::
                            fromLatin1("Quantitativo Discreto")));
102                 }
103                 break;
104             default:
105                 break;
106             }
107         }
108         else
109         {
110             combo->setCurrentIndex(combo->findText(QString::fromLatin1("")
                    ));
111         }
112
113         ui.tableWidgetEditarTiposAtributos->resizeColumnsToContents();
114     }
115
116     ui.pushButtonFim->setVisible(true);
117
118     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
119     ui.labelStatus->repaint();
120 }
121 void DialogImportacaoDados::atualizaTiposAtributos()
122 {
123     ui.labelStatus->setText(QString::fromLatin1("Salvando as alterações..."
            ));
124     ui.labelStatus->repaint();
125
126     _tipos.clear();
127
128     const int count = ui.tableWidgetEditarTiposAtributos->rowCount();
129
130     for (size_t i = 0; i < count; i++)
131     {
132         TipoDado_AT tipo;
133         QComboBox *combo = (QComboBox*) ui.tableWidgetEditarTiposAtributos
                ->cellWidget(i, COLUNA_TIPO_ATRIBUTO);
134
135         QString atual = combo->currentText();
136
137         if (atual == QString::fromLatin1("Qualitativo"))
138         {
139             tipo = QUALITATIVO;
140         }
141         else if (atual == QString::fromLatin1("Quantitativo Contínuo"))
142         {
143             tipo = QUANT_CONTINUO;
144         }
145         else if (atual == QString::fromLatin1("Quantitativo Discreto"))
146         {
147             tipo = QUANT_DISCRETO;
148         }
```

```
149
150         _tipos.push_back(tipo);
151     }
152
153     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
154     ui.labelStatus->repaint();
155 }
156 void DialogImportacaoDados::converterSetDados()
157 {
158     setupProgressBar();
159     _progress->setLabelText(QString::fromLatin1("Convertendo dados..."));
160     _progress->show();
161     QCoreApplication::processEvents(QEventLoop::ExcludeUserInputEvents);
162
163     _valoresColunas = vector<vector<ValorBase *const>>();
164     TabelaDeRegistros::dadosStringToValoresBases(_dadosColunasStr, _tipos,
            _valoresColunas, _progress);
165
166     _progress->deleteLater();
167     _progress->close();
168
169     iniciarColetarEstatisticas();
170 }
```

Listagem B.38: dialog_importacao_dados_pg_02_tipificar.cpp

```
1  #include "dialog_importacao_dados.h"
2
3  void DialogImportacaoDados::proxClickPgDefInterCatValidos()
4  {
5      bool sucesso = true;
6
7      //informações dos atributos quant
8      vector<string> atributosQuant = vector<string>();
9      vector<vector<TabelaDeRegistros::Intervalo>> interCadaAtribQuant =
           vector<vector<TabelaDeRegistros::Intervalo>>();
10     vector<bool> missingsQuantTmb = vector<bool>();
11     vector<bool> invalidosQuantTmb = vector<bool>();
12
13     //informaçoes dos atributos qual
14     vector<string> atributosQual = vector<string>();
15     vector<vector<ValorBase *const>> valCadaAtribQual = vector<vector<
           ValorBase *const>>();
16     vector<bool> missingsQualTmb = vector<bool>();
17     vector<bool> invalidosQualTmb = vector<bool>();
18
19     //informações todos atributos
20     vector<string> todosAtributos = vector<string>();
21     vector<bool> temRegPorAtrib = vector<bool>();
22     vector<string> problemasPorAtrib = vector<string>();
23     _regsDescPorAtrib = map<string, vector<size_t>>();
24
25     string mensagemErro = "Não foi possível avançar pelos problemas com os
           atributos:\n\n";
26     bool algumTemProblema = false;
27
28     vector<size_t> registrosCosiderados = vector<size_t>();
29     vector<size_t> registrosDescosiderados = vector<size_t>();
30
31     for (vector<WidgetIntervalosQuant*>::const_iterator citWidget =
           _widgetsIntervalosQuant.cbegin();
32         citWidget != _widgetsIntervalosQuant.cend();
33         citWidget++)
34     {
35         WidgetIntervalosQuant *widgetIntervalosQuant = *citWidget;
36
37         const string atributo = widgetIntervalosQuant->atributoAtual();
38
39         vector<TabelaDeRegistros::Intervalo> intervalos = vector<
               TabelaDeRegistros::Intervalo>();
40
41         try
42         {
43             widgetIntervalosQuant->gerarIntervalos(intervalos);
44         }
45         catch (std::exception &e)
46         {
47             const string msg = atributo +": " +e.what();
48             QMessageBox::warning(NULL, QString::fromLatin1("Aviso"),
                   QString::fromLatin1(msg.c_str()));
```

```
49              goto fim;
50          }
51
52          atributosQuant.push_back(atributo);
53          interCadaAtribQuant.push_back(intervalos);
54          todosAtributos.push_back(atributo);
55
56          vector<size_t> registrosCons = vector<size_t>();
57          vector<size_t> registrosDesc = vector<size_t>();
58
59          string problema = "";
60          temRegPorAtrib.push_back(widgetIntervalosQuant->haRegistro(problema
                , registrosCons));
61          problemasPorAtrib.push_back(problema);
62
63          TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
                registrosCons, registrosDesc);
64          _regsDescPorAtrib[atributo] = registrosDesc;
65
66          missingsQuantTmb.push_back(widgetIntervalosQuant->
                considerarMissings());
67          invalidosQuantTmb.push_back(widgetIntervalosQuant->
                considerarInvalidos());
68      }
69
70      for (vector<WidgetCategoriasQual*>::const_iterator citWidget =
             _widgetsCategoriasQual.cbegin();
71          citWidget != _widgetsCategoriasQual.cend();
72          citWidget++)
73      {
74          WidgetCategoriasQual *widgetCategoriasQual = *citWidget;
75
76          const string atributo = widgetCategoriasQual->atributoAtual();
77
78          vector<ValorBase *const> valores = vector<ValorBase *const>();
79          widgetCategoriasQual->obterSelecionados(valores);
80
81          atributosQual.push_back(atributo);
82          valCadaAtribQual.push_back(valores);
83          todosAtributos.push_back(atributo);
84
85          vector<size_t> registrosValidos = vector<size_t>();
86          vector<size_t> registrosInvalidos = vector<size_t>();
87
88          string problema = "";
89          temRegPorAtrib.push_back(widgetCategoriasQual->haRegistros(problema
                , registrosValidos));
90          problemasPorAtrib.push_back(problema);
91
92          TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
                registrosValidos, registrosInvalidos);
93          _regsDescPorAtrib[atributo] = registrosInvalidos;
94
95          missingsQualTmb.push_back(widgetCategoriasQual->considerarMissings
                ());
96          invalidosQualTmb.push_back(false);
97      }
98
99      for (size_t i = 0; i < todosAtributos.size(); i++)
100     {
101         const bool &temRegistro = temRegPorAtrib[i];
102
103         if (!temRegistro)
104         {
105             const string &atributo = todosAtributos[i];
106             const string &problema = problemasPorAtrib[i];
107
108             mensagemErro += "\t - " +atributo +": " +problema +"\n";
109
110             algumTemProblema = true;
111         }
112     }
113
114     if (algumTemProblema)
115     {
116         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1(mensagemErro.c_str()));
117         goto fim;
118     }
119
120     TabelaDeRegistros::obterInstancia()->identificarRegistrosMultiAtributos
            (atributosQuant, interCadaAtribQuant, missingsQuantTmb,
```

```
            invalidosQuantTmb, atributosQual, valCadaAtribQual,
            missingsQualTmb, invalidosQualTmb, registrosCosiderados);
121
122     if (registrosCosiderados.empty())
123     {
124         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("A combinação de intervalos/categorias resulta em
                nenhum registro!"));
125         goto fim;
126     }
127
128     TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
            registrosCosiderados, registrosDescosiderados);
129
130     if (!registrosDescosiderados.empty())
131     {
132         setupPgTratarForaIntervCat(registrosDescosiderados);
133         ui.stackedWidgetInformacoes->setCurrentIndex(
                PAGE_TRATAR_VALORES_INVALIDOS);
134     }
135     else
136     {
137         QMessageBox::information(NULL, QString::fromLatin1("Informação"),
                QString::fromLatin1("Todos as registros pertencem aos
                intervalos/categorias definidos."));
138         proxClickPgTratarForaIntervCat(); //pulando a pagina de tratar
                intervalos
139     }
140
141     fim:
142     for (vector<vector<ValorBase *const>>::const_iterator cit1 =
            valCadaAtribQual.cbegin();
143         cit1 != valCadaAtribQual.cbegin();
144         cit1++)
145     {
146         vector<ValorBase *const> valAtribQual = *cit1;
147         for (vector<ValorBase *const>::const_iterator cit2 = valAtribQual.
                cbegin();
148         cit2 != valAtribQual.cbegin();
149         cit2++)
150         {
151             delete *cit2;
152         }
153     }
154 }
155 void DialogImportacaoDados::voltarClickPgDefInterCatValidos()
156 {
157     ui.tabWidgetValoresValidos->clear();
158     _widgetsCategoriasQual.clear();
159     _widgetsIntervalosQuant.clear();
160
161     ui.stackedWidgetInformacoes->setCurrentIndex(PAGE_TIPIFICAR_ATRIBUTOS);
162 }
163 void DialogImportacaoDados::setupPgDefInterCatValidos()
164 {
165     const vector<string> &atributos = TabelaDeRegistros::obterInstancia()->
            atributos();
166
167     size_t iTab = 0;
168
169     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
170         citAtributo != atributos.cend();
171         citAtributo++, iTab++)
172     {
173         const string &atributo = *citAtributo;
174
175         if (TabelaDeRegistros::obterInstancia()->ehQuant(atributo))
176         {
177             WidgetIntervalosQuant *widgetIntervalosQuant = new
                    WidgetIntervalosQuant(atributo, ui.tabWidgetValoresValidos
                    );
178             widgetIntervalosQuant->setVisibilidadeComboAtributos(false);
179             widgetIntervalosQuant->setAtributoAtualComboBoxAtributos(
                    atributo);
180             widgetIntervalosQuant->checkConsiderarInvalidos(true);
181             widgetIntervalosQuant->checkConsiderarMissings(true);
182
183             _widgetsIntervalosQuant.push_back(widgetIntervalosQuant);
184
185             ui.tabWidgetValoresValidos->insertTab(iTab,
                    widgetIntervalosQuant, QString::fromLatin1(atributo.c_str
                    ()));
```

```
186             }
187             else if (TabelaDeRegistros::obterInstancia()->ehQualitativo(
                    atributo))
188             {
189                 WidgetCategoriasQual *widgetCategoriasQual = new
                        WidgetCategoriasQual(atributo, ui.tabWidgetValoresValidos)
                        ;
190                 widgetCategoriasQual->setVisibilidadeComboAtributos(false);
191                 widgetCategoriasQual->setAtributoAtualComboBoxAtributos(
                        atributo);
192                 widgetCategoriasQual->marcarTodos();
193                 widgetCategoriasQual->checkConsiderarMissings(true);
194
195                 _widgetsCategoriasQual.push_back(widgetCategoriasQual);
196
197                 ui.tabWidgetValoresValidos->insertTab(iTab,
                        widgetCategoriasQual, QString::fromLatin1(atributo.c_str()
                        ));
198             }
199         }
200 }
```

Listagem B.39: dialog_importacao_dados_pg_03_def_interv_cat.cpp

```
1  #include "dialog_importacao_dados.h"
2
3  #include "populador.h"
4
5  void DialogImportacaoDados::proxClickPgTratarForaIntervCat()
6  {
7      setupPgSelTecOutliers();
8
9      ui.stackedWidgetInformacoes->setCurrentIndex(PAGE_SEL_TEC_OUTLIERS);
10 }
11 void DialogImportacaoDados::voltarClickPgTratarForaIntervCat()
12 {
13     ui.widgetPgTratarForaInterCateg->layout()->removeWidget(
           _widgetTabelaEditoraTratarInterCateg);
14     if (_widgetTabelaEditoraTratarInterCateg) {delete
           _widgetTabelaEditoraTratarInterCateg;}
15     _widgetTabelaEditoraTratarInterCateg = NULL;
16
17     ui.stackedWidgetInformacoes->setCurrentIndex(PAGE_INTERV_CAR_VALIDOS);
18 }
19 void DialogImportacaoDados::setupPgTratarForaIntervCat(const vector<size_t>
       &todosIndicesFora)
20 {
21     _widgetTabelaEditoraTratarInterCateg = new WidgetTabelaEditora(
           todosIndicesFora, ui.widgetPgTratarForaInterCateg);
22     _widgetTabelaEditoraTratarInterCateg->destacar(_regsDescPorAtrib);
23     _widgetTabelaEditoraTratarInterCateg->setBarraMenuVisivel(false);
24     ui.widgetPgTratarForaInterCateg->layout()->addWidget(
           _widgetTabelaEditoraTratarInterCateg);
25
26     vector<string> strsComboBoxRemover = vector<string>();
27     strsComboBoxRemover.push_back("Selecionado");
28     strsComboBoxRemover.push_back("Todos destacados na coluna");
29     strsComboBoxRemover.push_back("Todos");
30
31     Populador::populeComboBox(ui.comboBoxRemoverFora, strsComboBoxRemover);
32
33     connect(ui.pushButtonRemoverFora, SIGNAL(clicked()), this, SLOT(
           removerForaClick()));
34 }
35
36 void DialogImportacaoDados::removerForaClick()
37 {
38     const int selecionadoIdx = 0;
39     const int todosDestacadosIdx = 1;
40     const int todosIdx = 2;
41
42     int atualIdx = ui.comboBoxRemoverFora->currentIndex();
43
44     switch (atualIdx)
45     {
46     case selecionadoIdx:
47         {
48             _widgetTabelaEditoraTratarInterCateg->removerLinha();
49         }
```

```
50          break;
51      case todosDestacadosIdx:
52          {
53              _widgetTabelaEditoraTratarInterCateg->
                    removerLinhasDestacadasDoAtributo();
54          }
55          break;
56      case todosIdx:
57          {
58              _widgetTabelaEditoraTratarInterCateg->removerTodasLinhas();
59          }
60          break;
61      }
62  }
```

Listagem B.40: dialog_importacao_dados_pg_04_tratar_fora_interv_cat.cpp

```
1   #include "dialog_importacao_dados.h"
2
3   #include "thread_worker_coletar_estatisticas.h"
4   #include "coletor_estatisticas.h"
5
6   #include <set>
7
8   void DialogImportacaoDados::proxClickPgSelTecOutliers()
9   {
10      const bool usarDiagramaCaixa = ui.checkBoxDiagramaCaixa->isChecked();
11      const bool usarZScore = ui.checkBoxZScore->isChecked();
12
13      if (!usarDiagramaCaixa && !usarZScore)
14      {
15          QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Selecione pelo menos uma técnica de detecção de
                outliers!"));
16          return;
17      }
18
19      setupProgressBar();
20      _progress->setLabelText(QString::fromLatin1("Detectando outliers..."));
21      _progress->show();
22      QCoreApplication::processEvents(QEventLoop::ExcludeUserInputEvents);
23
24      const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
            ()->atributosQuantitativos();
25      _indicesOutliersPorAtrib = map<string, vector<size_t>>();
26      set<size_t> todosIndicesOutliers = set<size_t>();
27
28      size_t iTab = 0;
29      for (vector<string>::const_iterator citAtributo = atributosQuant.cbegin
            ();
30          citAtributo != atributosQuant.cend();
31          citAtributo++, iTab++)
32      {
33          const string &atributo = *citAtributo;
34
35          map<size_t, ValorQuantitativo *const> mapIndiceOutlier = map<size_t
                , ValorQuantitativo *const>();
36
37          if (usarDiagramaCaixa)
38          {
39              const double fatorWI = _widgetsParamDiagCaixa[iTab]->fatorWI();
40              const double fatorWS = _widgetsParamDiagCaixa[iTab]->fatorWS();
41
42              ColetorEstatisticas::obterInstancia()->outliersDiagramaCaixa(
                    atributo, fatorWS, fatorWI, mapIndiceOutlier);
43          }
44
45          if (usarZScore)
46          {
47              const double zMin = _widgetsParamZscore[iTab]->zMin();
48              const double zMax = _widgetsParamZscore[iTab]->zMax();
49
50              ColetorEstatisticas::obterInstancia()->outliersZScore(atributo,
                    zMin, zMax, mapIndiceOutlier);
51          }
52
53          vector<size_t> indicesOutliers = vector<size_t>();
```

```
54          indicesOutliers.reserve(mapIndiceOutlier.size());
55
56          for (map<size_t, ValorQuantitativo *const>::const_iterator
                 citIndiceValor = mapIndiceOutlier.cbegin();
57               citIndiceValor != mapIndiceOutlier.cend();
58               citIndiceValor++)
59          {
60              const size_t indice = citIndiceValor->first;
61
62              indicesOutliers.push_back(indice);
63              todosIndicesOutliers.insert(indice);
64          }
65
66          _indicesOutliersPorAtrib[atributo] = indicesOutliers;
67
68          _progress->setValue(min((int)((iTab + 1) * 1.0 / atributosQuant.
                size() * 100), 99));
69          QCoreApplication::processEvents(QEventLoop::ExcludeUserInputEvents)
                ;
70      }
71
72      vector<size_t> todosIndicesOutliersVector = vector<size_t>();
73      copy(todosIndicesOutliers.begin(), todosIndicesOutliers.end(),
            back_inserter(todosIndicesOutliersVector));
74
75      _progress->deleteLater();
76      _progress->close();
77
78      setupPgTratarOutliers(todosIndicesOutliersVector);
79
80      ui.stackedWidgetInformacoes->setCurrentIndex(PAGE_TRATAR_OUTLIERS);
81  }
82  void DialogImportacaoDados::voltarClickPgSelTecOutliers()
83  {
84      if (_widgetTabelaEditoraTratarInterCateg)
85      {
86          ui.stackedWidgetInformacoes->setCurrentIndex(
                PAGE_TRATAR_VALORES_INVALIDOS);
87      }
88      else
89      {
90          ui.stackedWidgetInformacoes->setCurrentIndex(
                PAGE_INTERV_CAR_VALIDOS);
91      }
92  }
93  void DialogImportacaoDados::setupPgSelTecOutliers()
94  {
95      const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
            ()->atributosQuantitativos();
96
97      ui.tabWidgetParamsDiagCaixa->clear();
98      ui.tabWidgetParamsZScore->clear();
99
100     _widgetsParamDiagCaixa.clear();
101     _widgetsParamZscore.clear();
102
103     size_t iTab = 0;
104     for (vector<string>::const_iterator citAtributo = atributosQuant.cbegin
            ();
105          citAtributo != atributosQuant.cend();
106          citAtributo++, iTab++)
107     {
108         const string &atributo = *citAtributo;
109
110         WidgetParamDiagCaixa *widgetParamDiagCaixa = new
                WidgetParamDiagCaixa(true, atributo, ui.
                tabWidgetParamsDiagCaixa);
111         ui.tabWidgetParamsDiagCaixa->insertTab(iTab, widgetParamDiagCaixa,
                QString::fromLatin1(atributo.c_str()));
112         _widgetsParamDiagCaixa.push_back(widgetParamDiagCaixa);
113         connect(widgetParamDiagCaixa, SIGNAL(aplicarParamsDCTodos(const
                double, const double)), this, SLOT(aplicarParamsDCTodos(const
                double, const double)));
114
115         WidgetParamZscore *widgetParamZscore = new WidgetParamZscore(true,
                atributo, ui.tabWidgetParamsZScore);
116         ui.tabWidgetParamsZScore->insertTab(iTab, widgetParamZscore,
                QString::fromLatin1(atributo.c_str()));
117         _widgetsParamZscore.push_back(widgetParamZscore);
118         connect(widgetParamZscore, SIGNAL(aplicarParamsZSTodos(const double
                , const double)), this, SLOT(aplicarParamsZSTodos(const double
                , const double)));
```

```
119     }
120 }
121 void DialogImportacaoDados::aplicarParamsDCTodos(const double fatorWI,
        const double fatorWS)
122 {
123     for (vector<WidgetParamDiagCaixa*>::const_iterator
            citWidgetParamDiagCaixa = _widgetsParamDiagCaixa.cbegin();
124          citWidgetParamDiagCaixa != _widgetsParamDiagCaixa.cend();
125          citWidgetParamDiagCaixa++)
126     {
127         WidgetParamDiagCaixa *widgetParamDiagCaixa = *
                citWidgetParamDiagCaixa;
128         widgetParamDiagCaixa->setFatorWI(fatorWI);
129         widgetParamDiagCaixa->setFatorWS(fatorWS);
130     }
131 }
132 void DialogImportacaoDados::aplicarParamsZSTodos(const double zMin, const
        double zMax)
133 {
134     for (vector<WidgetParamZscore*>::const_iterator citWidgetParamZscore =
            _widgetsParamZscore.cbegin();
135          citWidgetParamZscore != _widgetsParamZscore.cend();
136          citWidgetParamZscore++)
137     {
138         WidgetParamZscore *widgetParamZscore = *citWidgetParamZscore;
139         widgetParamZscore->setZMin(zMin);
140         widgetParamZscore->setZMax(zMax);
141     }
142 }
```

Listagem B.41: dialog_importacao_dados_pg_05_tec_outliers.cpp

```
1  #include "dialog_importacao_dados.h"
2
3  #include "populador.h"
4
5  void DialogImportacaoDados::proxClickPgTratarOutliers()
6  {
7      setupPgTratarMissingsInvalidos();
8      ui.stackedWidgetInformacoes->setCurrentIndex(
           PAGE_TRATAR_MISSINGS_INVALIDOS);
9  }
10 void DialogImportacaoDados::voltarClickPgTratarOutliers()
11 {
12     ui.widgetPgTratarOutlier->layout()->removeWidget(
           _widgetTabelaEditoraTratarOutliers);
13     if (_widgetTabelaEditoraTratarOutliers) {delete
           _widgetTabelaEditoraTratarOutliers;}
14     _widgetTabelaEditoraTratarOutliers = NULL;
15
16     ui.stackedWidgetInformacoes->setCurrentIndex(PAGE_SEL_TEC_OUTLIERS);
17 }
18 void DialogImportacaoDados::setupPgTratarOutliers(const vector<size_t> &
       todosIndicesOutliers)
19 {
20     _widgetTabelaEditoraTratarOutliers = new WidgetTabelaEditora(
           todosIndicesOutliers, ui.widgetPgTratarOutlier);
21     _widgetTabelaEditoraTratarOutliers->destacar(_indicesOutliersPorAtrib);
22     _widgetTabelaEditoraTratarOutliers->setBarraMenuVisivel(false);
23     ui.widgetPgTratarOutlier->layout()->addWidget(
           _widgetTabelaEditoraTratarOutliers);
24
25     vector<string> strsComboBoxRemover = vector<string>();
26     strsComboBoxRemover.push_back("Selecionado");
27     strsComboBoxRemover.push_back("Todos destacados na coluna");
28     strsComboBoxRemover.push_back("Todos");
29
30     Populador::populeComboBox(ui.comboBoxRemoverOutlier,
           strsComboBoxRemover);
31
32     connect(ui.pushButtonRemoverOutlier, SIGNAL(clicked()), this, SLOT(
           removerOutlierClick()));
33     connect(ui.pushButtonSubstituirOutlier, SIGNAL(clicked()),
           _widgetTabelaEditoraTratarOutliers, SLOT(substituaClick()));
34 }
35
36 void DialogImportacaoDados::removerOutlierClick()
37 {
38     const int selecionadoIdx = 0;
```

```
39      const int todosDestacadosIdx = 1;
40      const int todosIdx = 2;
41
42      int atualIdx = ui.comboBoxRemoverOutlier->currentIndex();
43
44      switch (atualIdx)
45      {
46      case selecionadoIdx:
47          {
48              _widgetTabelaEditoraTratarOutliers->removerLinha();
49          }
50          break;
51      case todosDestacadosIdx:
52          {
53              _widgetTabelaEditoraTratarOutliers->
                    removerLinhasDestacadasDoAtributo();
54          }
55          break;
56      case todosIdx:
57          {
58              _widgetTabelaEditoraTratarOutliers->removerTodasLinhas();
59          }
60          break;
61      }
62  }
```

Listagem B.42: dialog_importacao_dados_pg_06_tratar_outliers.cpp

```
1   #include "dialog_importacao_dados.h"
2
3   #include "populador.h"
4   #include "coletor_estatisticas.h"
5
6   void DialogImportacaoDados::proxClickPgTratarMissingsInvalidos()
7   {
8       accept();
9   }
10  void DialogImportacaoDados::voltarClickPgTratarMissingsInvalidos()
11  {
12      ui.widgetPgTratarMissingsInvalidos->layout()->removeWidget(
            _widgetTabelaEditoraTratarMissingsInvalidos);
13      if (_widgetTabelaEditoraTratarMissingsInvalidos) {delete
            _widgetTabelaEditoraTratarMissingsInvalidos;}
14      _widgetTabelaEditoraTratarMissingsInvalidos = NULL;
15
16      ui.stackedWidgetInformacoes->setCurrentIndex(
            PAGE_TRATAR_MISSINGS_INVALIDOS);
17
18  }
19  void DialogImportacaoDados::setupPgTratarMissingsInvalidos()
20  {
21      const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
            ()->atributosQuantitativos();
22      Populador::populeComboBox(ui.comboBoxVariaveisReferencia,
            atributosQuant);
23
24      connect(ui.pushButtonGerarRegistrosMissings, SIGNAL(clicked()), this,
            SLOT(pushButtonGerarRegistrosMissings()));
25      connect(ui.checkBoxUtilizarVarRef, SIGNAL(toggled(bool)), ui.
            comboBoxVariaveisReferencia, SLOT(setEnabled(bool)));
26      connect(ui.checkBoxUtilizarVarRef, SIGNAL(toggled(bool)), ui.
            pushButtonGerarRegistrosMissings, SLOT(setEnabled(bool)));
27
28      vector<size_t> registrosMissInv = vector<size_t>();
29      map<string, vector<size_t>> missInvaPorAtributo = map<string, vector<
            size_t>>();
30
31      TabelaDeRegistros::obterInstancia()->
            identificarRegistrosMissingsInvalidos(registrosMissInv,
            missInvaPorAtributo);
32
33      _widgetTabelaEditoraTratarMissingsInvalidos = new WidgetTabelaEditora(
            registrosMissInv, ui.widgetPgTratarMissingsInvalidos);
34      _widgetTabelaEditoraTratarMissingsInvalidos->destacar(
            missInvaPorAtributo);
35      _widgetTabelaEditoraTratarMissingsInvalidos->setBarraMenuVisivel(false)
            ;
36      ui.widgetPgTratarMissingsInvalidos->layout()->addWidget(
            _widgetTabelaEditoraTratarMissingsInvalidos);
```

```
37
38      vector<string> strsComboBoxRemover = vector<string>();
39      strsComboBoxRemover.push_back("Selecionado");
40      strsComboBoxRemover.push_back("Todos destacados na coluna");
41      strsComboBoxRemover.push_back("Todos");
42
43      Populador::populeComboBox(ui.comboBoxRemoverMissInv,
            strsComboBoxRemover);
44
45      connect(ui.pushButtonRemoverMissInv, SIGNAL(clicked()), this, SLOT(
            removerMissInvClick()));
46      connect(ui.pushButtonSubstituirMissInv, SIGNAL(clicked()),
            _widgetTabelaEditoraTratarMissingsInvalidos, SLOT(substituaClick()
            ));
47  }
48
49  void DialogImportacaoDados::pushButtonGerarRegistrosMissings()
50  {
51      if (ui.checkBoxUtilizarVarRef->isChecked())
52      {
53          const string atributoReferencia = ui.comboBoxVariaveisReferencia->
                currentText().toLatin1().data();
54          vector<int> valoresFaltantes = vector<int>();
55          string msg = "";
56          const bool sucesso = TabelaDeRegistros::obterInstancia()->
                ehVariavelReferencia(atributoReferencia, valoresFaltantes, msg
                );
57
58          if(!sucesso)
59          {
60              QMessageBox::warning(NULL, QString::fromLatin1("Aviso"),
                    QString::fromLatin1(msg.c_str()));
61              return;
62          }
63
64          for (vector<int>::const_iterator citValorFaltante =
                valoresFaltantes.cbegin();
65              citValorFaltante != valoresFaltantes.cend();
66              citValorFaltante++)
67          {
68              const int valorFaltanteInt = *citValorFaltante;
69              const size_t indiceNovoReg = TabelaDeRegistros::obterInstancia
                    ()->novoRegistro();
70
71              ValorBase *const valorFaltante = TabelaDeRegistros::
                    gerarValorQuant(valorFaltanteInt);
72              TabelaDeRegistros::obterInstancia()->atualizeRegistro(
                    indiceNovoReg, atributoReferencia, valorFaltante);
73              delete valorFaltante;
74          }
75
76          const vector<string> &atributos = TabelaDeRegistros::obterInstancia
                ()->atributos();
77          ColetorEstatisticas::obterInstancia()->
                coletarEstatisticasDosAtributos(atributos);
78      }
79
80      ui.widgetPgTratarMissingsInvalidos->layout()->removeWidget(
            _widgetTabelaEditoraTratarMissingsInvalidos);
81      if (_widgetTabelaEditoraTratarMissingsInvalidos) {delete
            _widgetTabelaEditoraTratarMissingsInvalidos;}
82      _widgetTabelaEditoraTratarMissingsInvalidos = NULL;
83
84      vector<size_t> registrosMissInv = vector<size_t>();
85      map<string, vector<size_t>> missInvaPorAtributo = map<string, vector<
            size_t>>();
86
87      TabelaDeRegistros::obterInstancia()->
            identificarRegistrosMissingsInvalidos(registrosMissInv,
            missInvaPorAtributo);
88
89      _widgetTabelaEditoraTratarMissingsInvalidos = new WidgetTabelaEditora(
            registrosMissInv, ui.widgetPgTratarMissingsInvalidos);
90      _widgetTabelaEditoraTratarMissingsInvalidos->destacar(
            missInvaPorAtributo);
91      _widgetTabelaEditoraTratarMissingsInvalidos->setBarraMenuVisivel(false)
            ;
92      ui.widgetPgTratarMissingsInvalidos->layout()->addWidget(
            _widgetTabelaEditoraTratarMissingsInvalidos);
93  }
94
95  void DialogImportacaoDados::removerMissInvClick()
```

```
96 {
97     const int selecionadoIdx = 0;
98     const int todosDestacadosIdx = 1;
99     const int todosIdx = 2;
100
101    int atualIdx = ui.comboBoxRemoverMissInv->currentIndex();
102
103    switch (atualIdx)
104    {
105    case selecionadoIdx:
106        {
107            _widgetTabelaEditoraTratarMissingsInvalidos->removerLinha();
108        }
109        break;
110    case todosDestacadosIdx:
111        {
112            _widgetTabelaEditoraTratarMissingsInvalidos->
                   removerLinhasDestacadasDoAtributo();
113        }
114        break;
115    case todosIdx:
116        {
117            _widgetTabelaEditoraTratarMissingsInvalidos->removerTodasLinhas
                   ();
118        }
119        break;
120    }
121 }
```

Listagem B.43: dialog_importacao_dados_pg_07_tratar_missings_invalidos.cpp

```
1  #include "dialog_nova_coluna.h"
2
3  #include "tabela_registros.h"
4
5  #include <QMessageBox>
6  #include <QDebug>
7
8  DialogNovaColuna::DialogNovaColuna(QWidget *parent)
9      : QDialog(parent)
10 {
11     ui.setupUi(this);
12     ui.comboBoxTipo->addItem(QString::fromLatin1("Quantitativo contínuo"),
           (int)TipoDado_AT::QUANT_CONTINUO);
13     ui.comboBoxTipo->addItem(QString::fromLatin1("Quantitativo discreto"),
           (int)TipoDado_AT::QUANT_DISCRETO);
14     ui.comboBoxTipo->addItem(QString::fromLatin1("Qualitativo"), (int)
           TipoDado_AT::QUALITATIVO);
15
16     connect(ui.pushButtonInserir, SIGNAL(clicked()), this, SLOT(
           inserirClick()));
17     connect(ui.pushButtonCancelar, SIGNAL(clicked()), this, SLOT(reject()))
           ;
18
19     setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
20 }
21
22 DialogNovaColuna::~DialogNovaColuna()
23 {
24     const void * address = static_cast<const void*>(&(*this));
25     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
26 }
27
28 void DialogNovaColuna::inserirClick()
29 {
30     const string novoAtributo = ui.lineEditNome->text().toLatin1().data();
31     const TipoDado_AT tipo = (TipoDado_AT)(ui.comboBoxTipo->itemData(ui.
           comboBoxTipo->currentIndex()).toInt());
32
33     if (novoAtributo.empty())
34     {
35         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
               fromLatin1("Insira um nome válido!"));
36     }
37     else if (TabelaDeRegistros::obterInstancia()->existeAtributo(
           novoAtributo))
38     {
```

```
39          QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Um atributo com este nome já existe!"));
40      }
41      else
42      {
43          emit inserirColuna(novoAtributo, tipo);
44          accept();
45      }
46  }
```

Listagem B.44: dialog_nova_coluna.cpp

```
1  #ifndef DIALOG_NOVA_COLUNA_H
2  #define DIALOG_NOVA_COLUNA_H
3
4  #include "ui_dialog_nova_coluna.h"
5
6  #include "enums_at.h"
7
8  #include <string>
9
10 using namespace std;
11
12 class DialogNovaColuna : public QDialog
13 {
14     Q_OBJECT
15
16 public:
17     DialogNovaColuna(QWidget *parent = 0);
18     ~DialogNovaColuna();
19
20 private:
21     Ui::DialogNovaColuna ui;
22
23 private slots:
24     void inserirClick();
25
26 signals:
27     void inserirColuna(const string&, const TipoDado_AT);
28 };
29
30 #endif // DIALOG_NOVA_COLUNA_H
```

Listagem B.45: dialog_nova_coluna.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogNovaColuna</class>
4   <widget class="QDialog" name="DialogNovaColuna">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>186</width>
10     <height>132</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>Inserir Coluna</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout">
17    <item>
18     <widget class="QGroupBox" name="groupBox">
19      <property name="title">
20       <string>Inserir Coluna</string>
21      </property>
22      <layout class="QFormLayout" name="formLayout">
23       <item row="0" column="0">
24        <widget class="QLabel" name="label">
25         <property name="text">
26          <string>Nome:</string>
27         </property>
28        </widget>
29       </item>
30       <item row="0" column="1">
31        <widget class="QLineEdit" name="lineEditNome"/>
```

```
32          </item>
33          <item row="1" column="0">
34           <widget class="QLabel" name="label_2">
35            <property name="text">
36             <string>Tipo</string>
37            </property>
38           </widget>
39          </item>
40          <item row="1" column="1">
41           <widget class="QComboBox" name="comboBoxTipo"/>
42          </item>
43         </layout>
44        </widget>
45       </item>
46       <item>
47        <layout class="QHBoxLayout" name="horizontalLayout">
48         <item>
49          <spacer name="horizontalSpacer">
50           <property name="orientation">
51            <enum>Qt::Horizontal</enum>
52           </property>
53           <property name="sizeHint" stdset="0">
54            <size>
55             <width>40</width>
56             <height>20</height>
57            </size>
58           </property>
59          </spacer>
60         </item>
61         <item>
62          <widget class="QPushButton" name="pushButtonInserir">
63           <property name="text">
64            <string>Inserir</string>
65           </property>
66          </widget>
67         </item>
68         <item>
69          <widget class="QPushButton" name="pushButtonCancelar">
70           <property name="text">
71            <string>Cancelar</string>
72           </property>
73          </widget>
74         </item>
75        </layout>
76       </item>
77      </layout>
78     </widget>
79     <layoutdefault spacing="6" margin="11"/>
80     <resources/>
81     <connections/>
82    </ui>
```

Listagem B.46: dialog_nova_coluna.ui

```
1  #include "dialog_rem_ide_outliers.h"
2
3  #include "coletor_estatisticas.h"
4
5  #include <QMessageBox>
6  #include <QDebug>
7
8  DialogRemIdeOutliers::DialogRemIdeOutliers(const string &atribuito, QWidget
       *parent)
9      : QDialog(parent), _atributo(atribuito)
10 {
11     ui.setupUi(this);
12
13     _widgetParamDiagCaixa = new WidgetParamDiagCaixa(false, atribuito, this
           );
14     _widgetParamZscore = new WidgetParamZscore(false, atribuito, this);
15
16     ui.tabWidget->insertTab(AbasIndice::DIAG_CAIXA, _widgetParamDiagCaixa,
           QString::fromLatin1("Diagrama Caixa"));
17     ui.tabWidget->insertTab(AbasIndice::Z_SCORE, _widgetParamZscore,
           QString::fromLatin1("Z-Score"));
18
19     connect(ui.pushButtonRemover, SIGNAL(clicked()), this, SLOT(
           removerClick()));
20     connect(ui.pushButtonIdentificar, SIGNAL(clicked()), this, SLOT(
           identificarClick()));
```

```
21      connect(ui.pushButtonCancelar, SIGNAL(clicked()), this, SLOT(reject()))
            ;
22
23      setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
24  }
25
26  DialogRemIdeOutliers::~DialogRemIdeOutliers()
27  {
28      const void * address = static_cast<const void*>(&(*this));
29      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
30  }
31
32  map<size_t, ValorQuantitativo *const> DialogRemIdeOutliers::getOutliers()
        const
33  {
34      map<size_t, ValorQuantitativo *const> mapIndiceOutlier = map<size_t,
            ValorQuantitativo *const>();
35
36      if (ui.tabWidget->currentIndex() == AbasIndice::DIAG_CAIXA)
37      {
38          const double fatorWS = _widgetParamDiagCaixa->fatorWS();
39          const double fatorWI = _widgetParamDiagCaixa->fatorWI();
40          ColetorEstatisticas::obterInstancia()->outliersDiagramaCaixa(
                _atributo, fatorWS, fatorWI, mapIndiceOutlier);
41      }
42      else if (ui.tabWidget->currentIndex() == AbasIndice::Z_SCORE)
43      {
44          const double zMin = _widgetParamZscore->zMin();
45          const double zMax = _widgetParamZscore->zMax();
46          ColetorEstatisticas::obterInstancia()->outliersZScore(_atributo,
                zMin, zMax, mapIndiceOutlier);
47      }
48
49      return mapIndiceOutlier;
50  }
51
52  void DialogRemIdeOutliers::removerClick()
53  {
54      map<size_t, ValorQuantitativo *const> mapIndiceOutlier = getOutliers();
55
56      vector<ValorBase *const> outliers = vector<ValorBase *const>();
57      outliers.reserve(mapIndiceOutlier.size());
58      for (map<size_t, ValorQuantitativo *const>::const_iterator
            citIndiceOutlier = mapIndiceOutlier.cbegin();
59          citIndiceOutlier != mapIndiceOutlier.cend();
60          citIndiceOutlier++)
61      {
62          outliers.push_back(static_cast<ValorBase *const>(citIndiceOutlier->
                second));
63      }
64
65      if (!outliers.empty())
66      {
67          emit removerRegistrosPorValor(outliers, _atributo, false, false);
68          accept();
69      }
70      else
71      {
72          QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Nenhum outlier encontrado!"));
73      }
74  }
75
76  void DialogRemIdeOutliers::identificarClick()
77  {
78      map<size_t, ValorQuantitativo *const> mapIndiceOutlier = getOutliers();
79
80      vector<ValorBase *const> outliers = vector<ValorBase *const>();
81      outliers.reserve(mapIndiceOutlier.size());
82      for (map<size_t, ValorQuantitativo *const>::const_iterator
            citIndiceOutlier = mapIndiceOutlier.cbegin();
83          citIndiceOutlier != mapIndiceOutlier.cend();
84          citIndiceOutlier++)
85      {
86          outliers.push_back(static_cast<ValorBase *const>(citIndiceOutlier->
                second));
87      }
88
89      if (!outliers.empty())
90      {
91          emit identificarRegistrosPorValor(outliers, _atributo, false, false
                );
```

```
92      }
93      else
94      {
95          QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Nenhum outlier encontrado!"));
96      }
97
98  }
```

Listagem B.47: dialog_rem_ide_outliers.cpp

```
1  #ifndef TELA_REM_IDE_OUTLIERS_H
2  #define TELA_REM_IDE_OUTLIERS_H
3
4  #include "ui_dialog_rem_ide_outliers.h"
5
6  #include "widget_param_diag_caixa.h"
7  #include "widget_param_zscore.h"
8
9  #include "valor_quantitativo.h"
10
11 #include <map>
12
13 using namespace std;
14
15 class DialogRemIdeOutliers : public QDialog
16 {
17     Q_OBJECT
18
19 public:
20     DialogRemIdeOutliers(const string &atributo, QWidget *parent = 0);
21     ~DialogRemIdeOutliers();
22
23 private:
24     enum AbasIndice{DIAG_CAIXA = 0, Z_SCORE = 1};
25
26     Ui::DialogRemIdeOutliers ui;
27     WidgetParamDiagCaixa *_widgetParamDiagCaixa;
28     WidgetParamZscore *_widgetParamZscore;
29
30     string _atributo;
31
32     map<size_t, ValorQuantitativo *const> getOutliers() const;
33
34 private slots:
35     void removerClick();
36     void identificarClick();
37
38 signals:
39     void removerRegistrosPorValor(const vector<ValorBase *const>&, const
            string&, const bool, const bool);
40     void identificarRegistrosPorValor(const vector<ValorBase *const>&,
            const string&, const bool, const bool);
41 };
42
43 #endif // TELA_REM_IDE_OUTLIERS_H
```

Listagem B.48: dialog_rem_ide_outliers.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogRemIdeOutliers</class>
4   <widget class="QDialog" name="DialogRemIdeOutliers">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>574</width>
10     <height>358</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>Remover/Identidicar Outliers</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout_2">
17    <item>
```

```
18      <widget class="QGroupBox" name="groupBox">
19       <property name="title">
20        <string>Remover/Identificar Outliers</string>
21       </property>
22       <layout class="QVBoxLayout" name="verticalLayout">
23        <item>
24         <widget class="QTabWidget" name="tabWidget"/>
25        </item>
26       </layout>
27      </widget>
28     </item>
29     <item>
30      <layout class="QHBoxLayout" name="horizontalLayout">
31       <item>
32        <spacer name="horizontalSpacer">
33         <property name="orientation">
34          <enum>Qt::Horizontal</enum>
35         </property>
36         <property name="sizeHint" stdset="0">
37          <size>
38           <width>40</width>
39           <height>20</height>
40          </size>
41         </property>
42        </spacer>
43       </item>
44       <item>
45        <widget class="QPushButton" name="pushButtonRemover">
46         <property name="text">
47          <string>Remover</string>
48         </property>
49        </widget>
50       </item>
51       <item>
52        <widget class="QPushButton" name="pushButtonIdentificar">
53         <property name="text">
54          <string>Identificar</string>
55         </property>
56        </widget>
57       </item>
58       <item>
59        <widget class="QPushButton" name="pushButtonCancelar">
60         <property name="text">
61          <string>Cancelar</string>
62         </property>
63        </widget>
64       </item>
65      </layout>
66     </item>
67    </layout>
68   </widget>
69   <layoutdefault spacing="6" margin="11"/>
70   <resources/>
71   <connections/>
72  </ui>
```

Listagem B.49: dialog_rem_ide_outliers.ui

```
1  #include "dialog_rem_ide_quali.h"
2
3  #include <QMessageBox>
4  #include <QDebug>
5
6  DialogRemIdeQuali::DialogRemIdeQuali(
7      const string &atributo,
8      QWidget *parent)
9      : QDialog(parent)
10 {
11     ui.setupUi(this);
12
13     _atributo = atributo;
14     ui.labelAtributo->setText(QString::fromLatin1(atributo.c_str()));
15
16     _widgetCategoriasQual = new WidgetCategoriasQual(atributo, ui.groupBox)
           ;
17     _widgetCategoriasQual->setHabilitadoComboBoxAtributos(false);
18     _widgetCategoriasQual->setAtributoAtualComboBoxAtributos(atributo);
19     ui.groupBox->layout()->addWidget(_widgetCategoriasQual);
20
```

```
21      connect(ui.pushButtonRemover, SIGNAL(clicked()), this, SLOT(
            removerClick()));
22      connect(ui.pushButtonIdentificar, SIGNAL(clicked()), this, SLOT(
            identificarClick()));
23      connect(ui.pushButtonCancelar, SIGNAL(clicked()), this, SLOT(reject()))
            ;
24
25      setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
26  }
27
28  DialogRemIdeQuali::~DialogRemIdeQuali()
29  {
30      const void * address = static_cast<const void*>(&(*this));
31      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
32  }
33
34  void DialogRemIdeQuali::removerClick()
35  {
36      vector<ValorBase *const> selecionados = vector<ValorBase *const>();
37      _widgetCategoriasQual->obterSelecionados(selecionados);
38
39      if (!selecionados.empty())
40      {
41          emit removerRegistrosPorValor(selecionados, _atributo, false, false
                );
42
43          for (vector<ValorBase *const>::const_iterator citSelecionado =
                selecionados.cbegin();
44              citSelecionado != selecionados.cend();
45              citSelecionado++)
46          {
47              delete *citSelecionado;
48          }
49
50          accept();
51      }
52      else
53      {
54          QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Nenhum valor selecionado!"));
55      }
56  }
57
58  void DialogRemIdeQuali::identificarClick()
59  {
60      vector<ValorBase *const> selecionados = vector<ValorBase *const>();
61      _widgetCategoriasQual->obterSelecionados(selecionados);
62
63      if (!selecionados.empty())
64      {
65          emit identificarRegistrosPorValor(selecionados, _atributo,
                _widgetCategoriasQual->considerarMissings(), false);
66
67          for (vector<ValorBase *const>::const_iterator citSelecionado =
                selecionados.cbegin();
68              citSelecionado != selecionados.cend();
69              citSelecionado++)
70          {
71              delete *citSelecionado;
72          }
73      }
74      else
75      {
76          QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Nenhum valor selecionado!"));
77      }
78  }
```

Listagem B.50: dialog_rem_ide_quali.cpp

```
1  #ifndef DIALOG_REM_IDE_QUALI_H
2  #define DIALOG_REM_IDE_QUALI_H
3
4  #include "ui_dialog_rem_ide_quali.h"
5
6  #include "widget_categorias_qual.h"
7
8  using namespace std;
9
```

```
10 class DialogRemIdeQuali : public QDialog
11 {
12     Q_OBJECT
13
14 public:
15     DialogRemIdeQuali(
16         const string &atributo,
17         QWidget *parent = 0);
18     ~DialogRemIdeQuali();
19
20 private:
21     Ui::DialogRemIdeQuali ui;
22     WidgetCategoriasQual *_widgetCategoriasQual;
23     string _atributo;
24
25 private slots:
26     void removerClick();
27     void identificarClick();
28
29 signals:
30     void removerRegistrosPorValor(const vector<ValorBase *const>&, const
          string&, const bool, const bool);
31     void identificarRegistrosPorValor(const vector<ValorBase *const>&,
          const string&, const bool, const bool);
32 };
33
34 #endif // DIALOG_REM_IDE_QUALI_H
```

Listagem B.51: dialog_rem_ide_quali.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogRemIdeQuali</class>
4   <widget class="QDialog" name="DialogRemIdeQuali">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>396</width>
10     <height>312</height>
11    </rect>
12   </property>
13   <property name="sizePolicy">
14    <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
15     <horstretch>0</horstretch>
16     <verstretch>0</verstretch>
17    </sizepolicy>
18   </property>
19   <property name="windowTitle">
20    <string>Remover/Identificar Registros por Valor Qualitativo</string>
21   </property>
22   <layout class="QVBoxLayout" name="verticalLayout_2">
23    <item>
24     <layout class="QFormLayout" name="formLayout">
25      <item row="0" column="0">
26       <widget class="QLabel" name="label">
27        <property name="text">
28         <string>Atributo:</string>
29        </property>
30       </widget>
31      </item>
32      <item row="0" column="1">
33       <widget class="QLabel" name="labelAtributo">
34        <property name="text">
35         <string>TextLabel</string>
36        </property>
37       </widget>
38      </item>
39     </layout>
40    </item>
41    <item>
42     <widget class="QGroupBox" name="groupBox">
43      <property name="sizePolicy">
44       <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
45        <horstretch>0</horstretch>
46        <verstretch>0</verstretch>
47       </sizepolicy>
48      </property>
49      <property name="title">
```

```
50        <string>Remover/Identificar Registros por valor</string>
51       </property>
52       <layout class="QVBoxLayout" name="verticalLayout"/>
53      </widget>
54     </item>
55     <item>
56      <layout class="QHBoxLayout" name="horizontalLayout">
57       <item>
58        <spacer name="horizontalSpacer">
59         <property name="orientation">
60          <enum>Qt::Horizontal</enum>
61         </property>
62         <property name="sizeHint" stdset="0">
63          <size>
64           <width>40</width>
65           <height>20</height>
66          </size>
67         </property>
68        </spacer>
69       </item>
70       <item>
71        <widget class="QPushButton" name="pushButtonRemover">
72         <property name="text">
73          <string>Remover</string>
74         </property>
75        </widget>
76       </item>
77       <item>
78        <widget class="QPushButton" name="pushButtonIdentificar">
79         <property name="text">
80          <string>Identificar</string>
81         </property>
82        </widget>
83       </item>
84       <item>
85        <widget class="QPushButton" name="pushButtonCancelar">
86         <property name="text">
87          <string>Cancelar</string>
88         </property>
89        </widget>
90       </item>
91      </layout>
92     </item>
93    </layout>
94   </widget>
95   <resources/>
96   <connections/>
97  </ui>
```

Listagem B.52: dialog_rem_ide_quali.ui

```
1  #include "dialog_rem_ide_quant.h"
2  #include "utilidades.h"
3
4  #include <QDebug>
5
6  DialogRemIdeQuant::DialogRemIdeQuant(
7      const string &atributo,
8      const double maximo,
9      const double minimo,
10     QWidget *parent)
11     : QDialog(parent)
12 {
13     ui.setupUi(this);
14
15     _atributo = atributo;
16     _maximo = maximo;
17     _minimo = minimo;
18
19     ui.labelMaximoEdit->setText(QString::fromLatin1(Utilidades::dblToStdStr
           (maximo, 0).c_str()));
20     ui.labelMinimoEdit->setText(QString::fromLatin1(Utilidades::dblToStdStr
           (minimo, 0).c_str()));
21     ui.labelAtributo->setText(QString::fromLatin1(atributo.c_str()));
22
23     _widgetIntervalosQuant = new WidgetIntervalosQuant(atributo, ui.
           groupBoxValor);
24     _widgetIntervalosQuant->setHabilitadoComboBoxAtributos(false);
25     _widgetIntervalosQuant->setAtributoAtualComboBoxAtributos(atributo);
```

```
26
27      ui.groupBoxValor->layout()->addWidget(_widgetIntervalosQuant);
28
29      connect(ui.pushButtonIdentificar, SIGNAL(clicked()), this, SLOT(
            identificarClick()));
30      connect(ui.pushButtonRemover, SIGNAL(clicked()), this, SLOT(
            removerClick()));
31      connect(ui.pushButtonCancelar, SIGNAL(clicked()), this, SLOT(reject()))
            ;
32
33      setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
34  }
35
36  DialogRemIdeQuant::~DialogRemIdeQuant()
37  {
38      const void * address = static_cast<const void*>(&(*this));
39      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
40  }
41
42  void DialogRemIdeQuant::identificarClick()
43  {
44      const bool missingsTmb = _widgetIntervalosQuant->considerarMissings();
45      const bool invalidosTmb = _widgetIntervalosQuant->considerarInvalidos()
            ;
46
47      vector<TabelaDeRegistros::Intervalo> intervalos = vector<
            TabelaDeRegistros::Intervalo>();
48
49      try
50      {
51          _widgetIntervalosQuant->gerarIntervalos(intervalos);
52          emit identificarRegistrosQuantPorIntervalo(intervalos, _atributo,
                missingsTmb, invalidosTmb);
53      }
54      catch (std::exception &e)
55      {
56          QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1(e.what()));
57      }
58  }
59
60  void DialogRemIdeQuant::removerClick()
61  {
62      const bool missingsTmb = _widgetIntervalosQuant->considerarMissings();
63      const bool invalidosTmb = _widgetIntervalosQuant->considerarInvalidos()
            ;
64
65      vector<TabelaDeRegistros::Intervalo> intervalos = vector<
            TabelaDeRegistros::Intervalo>();
66
67      try
68      {
69          _widgetIntervalosQuant->gerarIntervalos(intervalos);
70          emit removerRegistrosQuantPorIntervalo(intervalos, _atributo,
                missingsTmb, invalidosTmb);
71          accept();
72      }
73      catch (std::exception &e)
74      {
75          QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1(e.what()));
76      }
77  }
```

Listagem B.53: dialog_rem_ide_quant.cpp

```
1  #ifndef DIALOG_REM_IDE_QUANT_H
2  #define DIALOG_REM_IDE_QUANT_H
3
4  #include "ui_dialog_rem_ide_quant.h"
5  #include "widget_intervalos_quant.h"
6
7  #include <QMessageBox>
8
9  using namespace std;
10
11 class DialogRemIdeQuant : public QDialog
12 {
13     Q_OBJECT
```

```
14
15 public:
16     DialogRemIdeQuant(
17         const string &atributo,
18         const double maximo,
19         const double minimo,
20         QWidget *parent = 0);
21     ~DialogRemIdeQuant();
22
23 private:
24     Ui::DialogRemIdeQuant ui;
25     WidgetIntervalosQuant *_widgetIntervalosQuant;
26     string _atributo;
27     double _maximo;
28     double _minimo;
29
30
31
32
33 private slots:
34     void identificarClick();
35     void removerClick();
36
37 signals:
38     void removerRegistrosPorValor(const vector<ValorBase *const>&, const
           string&, const bool, const bool);
39     void removerRegistrosQuantPorIntervalo(const vector<TabelaDeRegistros::
           Intervalo>&, const string&, const bool, const bool);
40     void identificarRegistrosPorValor(const vector<ValorBase *const>&,
           const string&, const bool, const bool);
41     void identificarRegistrosQuantPorIntervalo(const vector<
           TabelaDeRegistros::Intervalo>&, const string&, const bool, const
           bool);
42 };
43
44 #endif // DIALOG_REM_IDE_QUANT_H
```

Listagem B.54: dialog_rem_ide_quant.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogRemIdeQuant</class>
4   <widget class="QDialog" name="DialogRemIdeQuant">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>537</width>
10     <height>351</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>Remover/Identificar Registros por Valor Quantitativo</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout">
17    <item>
18     <layout class="QFormLayout" name="formLayout">
19      <property name="fieldGrowthPolicy">
20       <enum>QFormLayout::AllNonFixedFieldsGrow</enum>
21      </property>
22      <item row="1" column="0">
23       <widget class="QLabel" name="label_2">
24        <property name="text">
25         <string>Mínimo:</string>
26        </property>
27       </widget>
28      </item>
29      <item row="1" column="1">
30       <widget class="QLabel" name="labelMinimoEdit">
31        <property name="text">
32         <string>TextLabel</string>
33        </property>
34       </widget>
35      </item>
36      <item row="2" column="0">
37       <widget class="QLabel" name="label_5">
38        <property name="text">
39         <string>Máximo:</string>
40        </property>
```

```
41          </widget>
42         </item>
43         <item row="2" column="1">
44          <widget class="QLabel" name="labelMaximoEdit">
45           <property name="text">
46            <string>TextLabel</string>
47           </property>
48          </widget>
49         </item>
50         <item row="0" column="0">
51          <widget class="QLabel" name="label">
52           <property name="text">
53            <string>Atributo:</string>
54           </property>
55          </widget>
56         </item>
57         <item row="0" column="1">
58          <widget class="QLabel" name="labelAtributo">
59           <property name="text">
60            <string>TextLabel</string>
61           </property>
62          </widget>
63         </item>
64        </layout>
65       </item>
66       <item>
67        <widget class="QGroupBox" name="groupBoxValor">
68         <property name="sizePolicy">
69          <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
70           <horstretch>0</horstretch>
71           <verstretch>0</verstretch>
72          </sizepolicy>
73         </property>
74         <property name="title">
75          <string>Valor</string>
76         </property>
77         <layout class="QVBoxLayout" name="verticalLayout_2"/>
78        </widget>
79       </item>
80       <item>
81        <layout class="QHBoxLayout" name="horizontalLayout_2">
82         <item>
83          <spacer name="horizontalSpacer">
84           <property name="orientation">
85            <enum>Qt::Horizontal</enum>
86           </property>
87           <property name="sizeHint" stdset="0">
88            <size>
89             <width>40</width>
90             <height>20</height>
91            </size>
92           </property>
93          </spacer>
94         </item>
95         <item>
96          <widget class="QPushButton" name="pushButtonRemover">
97           <property name="text">
98            <string>Remover</string>
99           </property>
100         </widget>
101        </item>
102        <item>
103         <widget class="QPushButton" name="pushButtonIdentificar">
104          <property name="text">
105           <string>Identificar</string>
106          </property>
107         </widget>
108        </item>
109        <item>
110         <widget class="QPushButton" name="pushButtonCancelar">
111          <property name="text">
112           <string>Cancelar</string>
113          </property>
114         </widget>
115        </item>
116       </layout>
117      </item>
118     </layout>
119    </widget>
120    <resources/>
121    <connections/>
122   </ui>
```

Listagem B.55: dialog_rem_ide_quant.ui

```
#include "dialog_renomear.h"

#include "tabela_registros.h"

#include <QDebug>

using namespace std;

DialogRenomear::DialogRenomear(
    const string &atributo,
    QWidget *parent)
    : QDialog(parent)
{
    ui.setupUi(this);

    connect(ui.pushButtonRenomear, SIGNAL(clicked()), this, SLOT(
        renomearClique()));
    connect(ui.pushButtonCancelar, SIGNAL(clicked()), this, SLOT(reject()))
        ;

    _atributo = atributo;

    ui.labelNomeAtualEdit->setText(QString::fromLatin1(_atributo.c_str()));

    setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
}

DialogRenomear::~DialogRenomear()
{
    const void * address = static_cast<const void*>(&(*this));
    qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
}

void DialogRenomear::renomearClique()
{
    const string novoNome = (string) ui.lineEditNovoNome->text().toLatin1()
        .data();

    if (novoNome.empty())
    {
        QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
            fromLatin1("Insira um novo nome!"));
    }
    else if (TabelaDeRegistros::obterInstancia()->existeAtributo(novoNome))
    {
        QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
            fromLatin1("Um atributo com este nome já existe!"));
    }
    else
    {
        emit renomear(_atributo, novoNome);
        accept();
    }
}
```

Listagem B.56: dialog_renomear.cpp

```
#ifndef DIALOG_RENOMEAR_H
#define DIALOG_RENOMEAR_H

#include "ui_dialog_renomear.h"

#include <QMessageBox>

#include <string>

using namespace std;

class DialogRenomear : public QDialog
{
    Q_OBJECT

```

```
16 public:
17     DialogRenomear(
18         const string &atributo,
19         QWidget *parent = 0);
20     ~DialogRenomear();
21
22 private:
23     Ui::DialogRenomear ui;
24     string _atributo;
25
26 private slots:
27     void renomearClique();
28
29 signals:
30     void renomear(const string&, const string&);
31 };
32
33 #endif // DIALOG_RENOMEAR_H
```

Listagem B.57: dialog_renomear.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogRenomear</class>
4   <widget class="QDialog" name="DialogRenomear">
5    <property name="enabled">
6     <bool>true</bool>
7    </property>
8    <property name="geometry">
9     <rect>
10     <x>0</x>
11     <y>0</y>
12     <width>360</width>
13     <height>140</height>
14    </rect>
15   </property>
16   <property name="sizePolicy">
17    <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
18     <horstretch>0</horstretch>
19     <verstretch>0</verstretch>
20    </sizepolicy>
21   </property>
22   <property name="minimumSize">
23    <size>
24     <width>360</width>
25     <height>140</height>
26    </size>
27   </property>
28   <property name="maximumSize">
29    <size>
30     <width>360</width>
31     <height>140</height>
32    </size>
33   </property>
34   <property name="windowTitle">
35    <string>Renomear Atributo</string>
36   </property>
37   <layout class="QVBoxLayout" name="verticalLayout_3">
38    <item>
39     <layout class="QVBoxLayout" name="verticalLayout_2">
40      <item>
41       <widget class="QGroupBox" name="groupBox">
42        <property name="title">
43         <string>Renomear Atributo</string>
44        </property>
45        <layout class="QVBoxLayout" name="verticalLayout">
46         <item>
47          <layout class="QFormLayout" name="formLayout">
48           <item row="0" column="0">
49            <widget class="QLabel" name="labelNomeAtual">
50             <property name="text">
51              <string>Nome Atual:</string>
52             </property>
53            </widget>
54           </item>
55           <item row="0" column="1">
56            <widget class="QLabel" name="labelNomeAtualEdit">
57             <property name="text">
58              <string>nomeAtual</string>
```

```
59              </property>
60             </widget>
61            </item>
62            <item row="1" column="0">
63             <widget class="QLabel" name="labelNovoNome">
64              <property name="text">
65               <string>Novo nome:</string>
66              </property>
67             </widget>
68            </item>
69            <item row="1" column="1">
70             <widget class="QLineEdit" name="lineEditNovoNome"/>
71            </item>
72           </layout>
73          </item>
74         </layout>
75        </widget>
76       </item>
77       <item>
78        <layout class="QHBoxLayout" name="horizontalLayout">
79         <item>
80          <spacer name="horizontalSpacer">
81           <property name="orientation">
82            <enum>Qt::Horizontal</enum>
83           </property>
84           <property name="sizeHint" stdset="0">
85            <size>
86             <width>40</width>
87             <height>20</height>
88            </size>
89           </property>
90          </spacer>
91         </item>
92         <item>
93          <widget class="QPushButton" name="pushButtonRenomear">
94           <property name="text">
95            <string>Renomear</string>
96           </property>
97          </widget>
98         </item>
99         <item>
100         <widget class="QPushButton" name="pushButtonCancelar">
101          <property name="enabled">
102           <bool>true</bool>
103          </property>
104          <property name="text">
105           <string>Cancelar</string>
106          </property>
107         </widget>
108        </item>
109       </layout>
110      </item>
111     </layout>
112    </item>
113   </layout>
114  </widget>
115  <layoutdefault spacing="6" margin="11"/>
116  <resources/>
117  <connections/>
118 </ui>
```

Listagem B.58: dialog_renomear.ui

```
1  #include "dialog_subst_qual.h"
2
3  #include "tabela_registros.h"
4  #include "populador.h"
5
6  #include <QMessageBox>
7  #include <QDebug>
8
9  DialogSubstQual::DialogSubstQual(
10     const vector<string> &categoriasAtributoQualitativo,
11     const string &atributo,
12     QWidget *parent)
13     : QDialog(parent)
14 {
15     ui.setupUi(this);
16
```

```
17      _atributo = atributo;
18      _widgetCategoriasQual = new WidgetCategoriasQual(atributo, ui.
            widgetLocalizar);
19      ui.widgetLocalizar->layout()->addWidget(_widgetCategoriasQual);
20
21      ui.labelAtributo->setText(QString::fromLatin1(_atributo.c_str()));
22
23      _widgetCategoriasQual->setEnabled(false);
24      _widgetCategoriasQual->setHabilitadoComboBoxAtributos(false);
25
26      connect(ui.pushButtonSubstituir, SIGNAL(clicked()), this, SLOT(
            subtituirClique()));
27      connect(ui.pushButtonCancelar, SIGNAL(clicked()), this, SLOT(reject()))
            ;
28      connect(ui.radioButtonLocalizar, SIGNAL(toggled(bool)),
            _widgetCategoriasQual, SLOT(setEnabled(bool)));
29      connect(ui.radioButtonValorExistente, SIGNAL(toggled(bool)), ui.
            comboBoxSubstPor, SLOT(setEnabled(bool)));
30      connect(ui.radioButtonNovoValor, SIGNAL(toggled(bool)), ui.
            lineEditNovoValor, SLOT(setEnabled(bool)));
31
32      Populador::populeComboBox(ui.comboBoxSubstPor,
            categoriasAtributoQualitativo);
33
34      setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
35  }
36
37  DialogSubstQual::~DialogSubstQual()
38  {
39      const void * address = static_cast<const void*>(&(*this));
40      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
41  }
42
43  void DialogSubstQual::somenteDestacados(const bool ok)
44  {
45      ui.radioButtonLocalizar->setEnabled(!ok);
46      ui.radioButtonMissing->setEnabled(!ok);
47      ui.radioButtonTodos->setEnabled(!ok);
48      ui.radioButtonLocalizar->setEnabled(!ok);
49
50      ui.radioButtonDestacados->setChecked(ok);
51      ui.radioButtonDestacados->setEnabled(ok);
52  }
53
54  string DialogSubstQual::getNovoValor() const
55  {
56      string novoValor = "";
57
58      if (ui.radioButtonValorExistente->isChecked())
59      {
60          novoValor = ui.comboBoxSubstPor->currentText().toLatin1().data();
61      }
62      else if (ui.radioButtonNovoValor->isChecked())
63      {
64          novoValor = ui.lineEditNovoValor->text().toLatin1().data();
65      }
66
67      return novoValor;
68  }
69
70
71  void DialogSubstQual::subtituirClique()
72  {
73      bool sucesso = true;
74
75      const string valorNovoStr = getNovoValor();
76
77      ValorBase *valorNovo = TabelaDeRegistros::obterInstancia()->gerarValor(
            _atributo, valorNovoStr);
78      vector<ValorBase *const> valoresAntigos = vector<ValorBase *const>();
79
80      if (ui.radioButtonMissing->isChecked())
81      {
82          emit substituaMissings(_atributo, valorNovo);
83      }
84      else if (ui.radioButtonTodos->isChecked())
85      {
86          emit substituaTodos(_atributo, valorNovo);
87      }
88      else if (ui.radioButtonLocalizar->isChecked())
89      {
90          _widgetCategoriasQual->obterSelecionados(valoresAntigos);
```

```
91
92          if (!valoresAntigos.empty())
93          {
94              const bool missingsTmb = _widgetCategoriasQual->
                    considerarMissings();
95
96              emit substituaLocalizados(_atributo, valoresAntigos, valorNovo,
                     missingsTmb, false);
97          }
98          else
99          {
100             QMessageBox::warning(NULL, QString::fromLatin1("Aviso"),
                    QString::fromLatin1("Selecione pelo menos um valor para
                    ser localizado!"));
101             sucesso = false;
102         }
103     }
104     else if (ui.radioButtonDestacados->isChecked())
105     {
106         emit substituaDestacados(_atributo, valorNovo);
107     }
108
109     delete valorNovo;
110     for (vector<ValorBase *const>::const_iterator citValorAntigo =
            valoresAntigos.cbegin();
111         citValorAntigo != valoresAntigos.cend();
112         citValorAntigo++)
113     {
114         delete *citValorAntigo;
115     }
116
117     if (sucesso)
118     {
119         accept();
120     }
121 }
```

Listagem B.59: dialog_subst_qual.cpp

```
1  #ifndef DIALOG_SUBST_QUAL_H
2  #define DIALOG_SUBST_QUAL_H
3
4  #include "ui_dialog_subst_qual.h"
5
6  #include "widget_categorias_qual.h"
7
8  using namespace std;
9
10 class DialogSubstQual : public QDialog
11 {
12     Q_OBJECT
13
14 public:
15     DialogSubstQual(
16         const vector<string> &categoriasAtributoQualitativo,
17         const string &atributo,
18         QWidget *parent = 0);
19     ~DialogSubstQual();
20
21     void somenteDestacados(const bool ok);
22
23 private:
24     Ui::DialogSubstQual ui;
25     WidgetCategoriasQual *_widgetCategoriasQual;
26     string _atributo;
27
28     string getNovoValor() const;
29
30 private slots:
31     void subtituirClique();
32
33 signals:
34     void substituaMissings(const string&, ValorBase *const);
35     void substituaTodos(const string&, ValorBase*);
36     void substituaDestacados(const string&, ValorBase*);
37     void substituaLocalizados(const string&, const vector<ValorBase *const
           >&, ValorBase *const, const bool missingsTmb, const bool
           invalidosTmb);
38 };
```

```
39
40 #endif // DIALOG_SUBST_QUAL_H
```

Listagem B.60: dialog_subst_qual.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogSubstQual</class>
4   <widget class="QDialog" name="DialogSubstQual">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>376</width>
10     <height>445</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>Substituir Qualitativo</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout_2">
17    <item>
18     <layout class="QFormLayout" name="formLayout_2">
19      <item row="0" column="0">
20       <widget class="QLabel" name="label">
21        <property name="text">
22         <string>Atributo:</string>
23        </property>
24       </widget>
25      </item>
26      <item row="0" column="1">
27       <widget class="QLabel" name="labelAtributo">
28        <property name="text">
29         <string>TextLabel</string>
30        </property>
31       </widget>
32      </item>
33     </layout>
34    </item>
35    <item>
36     <widget class="QGroupBox" name="groupBox">
37      <property name="title">
38       <string>Substituir Valores</string>
39      </property>
40      <layout class="QVBoxLayout" name="verticalLayout">
41       <item>
42        <layout class="QHBoxLayout" name="horizontalLayout">
43         <item>
44          <widget class="QRadioButton" name="radioButtonMissing">
45           <property name="text">
46            <string>Missings</string>
47           </property>
48           <property name="checked">
49            <bool>true</bool>
50           </property>
51          </widget>
52         </item>
53         <item>
54          <widget class="QRadioButton" name="radioButtonTodos">
55           <property name="text">
56            <string>Todos</string>
57           </property>
58          </widget>
59         </item>
60         <item>
61          <widget class="QRadioButton" name="radioButtonLocalizar">
62           <property name="text">
63            <string>Localizar</string>
64           </property>
65          </widget>
66         </item>
67         <item>
68          <widget class="QRadioButton" name="radioButtonDestacados">
69           <property name="enabled">
70            <bool>true</bool>
71           </property>
72           <property name="text">
73            <string>Destacados</string>
74           </property>
```

```
75            </widget>
76           </item>
77          </layout>
78         </item>
79        </layout>
80       </widget>
81      </item>
82      <item>
83       <widget class="QWidget" name="widgetLocalizar" native="true">
84        <property name="sizePolicy">
85         <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
86          <horstretch>0</horstretch>
87          <verstretch>0</verstretch>
88         </sizepolicy>
89        </property>
90        <layout class="QVBoxLayout" name="verticalLayout_3">
91         <property name="leftMargin">
92          <number>0</number>
93         </property>
94         <property name="topMargin">
95          <number>0</number>
96         </property>
97         <property name="rightMargin">
98          <number>0</number>
99         </property>
100        <property name="bottomMargin">
101         <number>0</number>
102        </property>
103       </layout>
104      </widget>
105     </item>
106     <item>
107      <widget class="QGroupBox" name="groupBox_2">
108       <property name="minimumSize">
109        <size>
110         <width>0</width>
111         <height>0</height>
112        </size>
113       </property>
114       <property name="title">
115        <string>Substituir por:</string>
116       </property>
117       <layout class="QFormLayout" name="formLayout">
118        <item row="0" column="0">
119         <widget class="QRadioButton" name="radioButtonValorExistente">
120          <property name="text">
121           <string>Valor existente:</string>
122          </property>
123          <property name="checked">
124           <bool>true</bool>
125          </property>
126         </widget>
127        </item>
128        <item row="0" column="1">
129         <widget class="QComboBox" name="comboBoxSubstPor"/>
130        </item>
131        <item row="1" column="0">
132         <widget class="QRadioButton" name="radioButtonNovoValor">
133          <property name="enabled">
134           <bool>true</bool>
135          </property>
136          <property name="text">
137           <string>Novo valor:</string>
138          </property>
139         </widget>
140        </item>
141        <item row="1" column="1">
142         <widget class="QLineEdit" name="lineEditNovoValor">
143          <property name="enabled">
144           <bool>false</bool>
145          </property>
146         </widget>
147        </item>
148       </layout>
149      </widget>
150     </item>
151     <item>
152      <layout class="QHBoxLayout" name="horizontalLayout_2">
153       <item>
154        <spacer name="horizontalSpacer">
155         <property name="orientation">
156          <enum>Qt::Horizontal</enum>
```

```
157          </property>
158          <property name="sizeHint" stdset="0">
159           <size>
160            <width>40</width>
161            <height>20</height>
162           </size>
163          </property>
164         </spacer>
165        </item>
166        <item>
167         <widget class="QPushButton" name="pushButtonSubstituir">
168          <property name="text">
169           <string>Substituir</string>
170          </property>
171         </widget>
172        </item>
173        <item>
174         <widget class="QPushButton" name="pushButtonCancelar">
175          <property name="text">
176           <string>Cancelar</string>
177          </property>
178         </widget>
179        </item>
180       </layout>
181      </item>
182     </layout>
183    </widget>
184    <layoutdefault spacing="6" margin="11"/>
185    <resources/>
186    <connections/>
187   </ui>
```

Listagem B.61: dialog_subst_qual.ui

```
1  #include "dialog_subst_quant.h"
2
3  #include <QDoubleValidator>
4  #include <QMessageBox>
5  #include <QDebug>
6
7  #include "utilidades.h"
8  #include "coletor_estatisticas.h"
9  #include "populador.h"
10
11 DialogSubstQuant::DialogSubstQuant(
12     const string &atributo,
13     const vector<string> &atributosQuant,
14     const vector<vector<size_t>> &indicesValidosPorAtribQuant,
15     QWidget *parent)
16     : QDialog(parent), _atributo(atributo), _widgetEscolherDP(NULL)
17 {
18     ui.setupUi(this);
19
20     const ColetorEstatisticas::EstatisticaQuant &estatistica =
           ColetorEstatisticas::obterInstancia()->
           estatisticaAtributoQuantitativo(atributo);
21     _media = estatistica.media;
22
23     ui.labelAtributo->setText(QString::fromLatin1(_atributo.c_str()));
24
25     ui.radioButtonMissing->setChecked(true);
26     ui.radioButtoZero->setCheckable(true);
27
28     if (isnan(_media))
29     {
30         ui.radioButtonMedia->setEnabled(false);
31     }
32
33     const vector<string> &atributos = TabelaDeRegistros::obterInstancia()->
           atributos();
34     Populador::populeComboBox(ui.comboBoxAtributoMedParcial, atributos);
35
36     const size_t indiceAtributoDP = std::find(atributosQuant.begin(),
           atributosQuant.end(), atributo) - atributosQuant.begin();
37     const vector<size_t> &indicesValidosDP = indicesValidosPorAtribQuant[
           indiceAtributoDP];
38
39     setupIntervalosLocalizar();
40     setupMediaParcial();
```

```
41      setupRLS(atributosQuant, indicesValidosPorAtribQuant);
42      setupEscolherDP(indicesValidosDP);
43
44      connect(ui.comboBoxAtributoMedParcial, SIGNAL(currentIndexChanged(const
              QString&)), this, SLOT(
            currentIndexChangedComboBoxAtributoMedParcial()));
45      connect(ui.radioButtonIntervaloParaSubs, SIGNAL(toggled(bool)), ui.
            groupBoxIntervalosSubst, SLOT(setVisible(bool)));
46
47      connect(ui.radioButtoZero, SIGNAL(clicked(bool)), this, SLOT(
            radioButtonAlterado()));
48      connect(ui.radioButtonMedia, SIGNAL(clicked(bool)), this, SLOT(
            radioButtonAlterado()));
49      connect(ui.radioButtonMediaParcial, SIGNAL(clicked(bool)), this, SLOT(
            radioButtonAlterado()));
50      connect(ui.radioButtonDP, SIGNAL(clicked(bool)), this, SLOT(
            radioButtonAlterado()));
51      connect(ui.radioButtonRLS, SIGNAL(clicked(bool)), this, SLOT(
            radioButtonAlterado()));
52
53      connect(ui.pushButtonCalcMediaParcial, SIGNAL(clicked()), this, SLOT(
            calcularMediaParcialClique()));
54      connect(ui.pushButtonSubstituir, SIGNAL(clicked()), this, SLOT(
            subtituirClique()));
55      connect(ui.pushButtonCancelar, SIGNAL(clicked()), this, SLOT(reject()))
            ;
56
57      setWindowFlags(windowFlags() & ~Qt::WindowContextHelpButtonHint);
58
59      radioButtonAlterado();
60      currentIndexChangedComboBoxAtributoMedParcial();
61  }
62
63  DialogSubstQuant::~DialogSubstQuant()
64  {
65      const void * address = static_cast<const void*>(&(*this));
66      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
67  }
68
69  void DialogSubstQuant::somenteDestacados(const bool ok)
70  {
71      ui.radioButtonIntervaloParaSubs->setEnabled(!ok);
72      ui.radioButtonMissing->setEnabled(!ok);
73      ui.radioButtonTodos->setEnabled(!ok);
74      ui.radioButtonInvalidos->setEnabled(!ok);
75
76      ui.radioButtonDestacados->setChecked(ok);
77      ui.radioButtonDestacados->setEnabled(ok);
78  }
79
80  void DialogSubstQuant::radioButtonAlterado()
81  {
82      ui.stackedWidgetMediaParcialDpRLS->setVisible(ui.
            radioButtonMediaParcial->isChecked() || ui.radioButtonDP->
            isChecked() || ui.radioButtonRLS->isChecked());
83      ui.doubleSpinBoxValor->setEnabled(!ui.radioButtonDP->isChecked());
84
85      if (ui.radioButtoZero->isChecked())
86      {
87          ui.doubleSpinBoxValor->setValue(0);
88      }
89      else if (ui.radioButtonMedia->isChecked())
90      {
91          ui.doubleSpinBoxValor->setValue(_media);
92      }
93      else if (ui.radioButtonMediaParcial->isChecked())
94      {
95          ui.stackedWidgetMediaParcialDpRLS->setCurrentWidget(ui.
                pageMediaParcial);
96      }
97      else if (ui.radioButtonDP->isChecked())
98      {
99          ui.stackedWidgetMediaParcialDpRLS->setCurrentWidget(ui.pageDP);
100     }
101     else if (ui.radioButtonRLS->isChecked())
102     {
103         ui.stackedWidgetMediaParcialDpRLS->setCurrentWidget(ui.pageRLS);
104     }
105 }
106
107 void DialogSubstQuant::setupIntervalosLocalizar()
108 {
```

```
109         _widgetIntervalosQuantParaSubstituir = new WidgetIntervalosQuant(
                _atributo, ui.groupBoxIntervalosSubst);
110         _widgetIntervalosQuantParaSubstituir->setVisibilidadeComboAtributos(
                false); //é escondido pois só é considerado o atributo que se quer
                 substituir
111         ui.groupBoxIntervalosSubst->layout()->addWidget(
                _widgetIntervalosQuantParaSubstituir);
112         ui.groupBoxIntervalosSubst->setVisible(ui.radioButtonIntervaloParaSubs
                ->isChecked());
113     }
114
115     void DialogSubstQuant::setupMediaParcial()
116     {
117         const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
                ()->atributosQuantitativos();
118         const vector<string> atributosQual = TabelaDeRegistros::obterInstancia
                ()->atributosQualitativos();
119
120         _widgetIntervalosQuantMediaParcial = new WidgetIntervalosQuant(
                atributosQuant, ui.stackedWidgetIntervalosMediaParcial->widget(0))
                ;
121         _widgetIntervalosQuantMediaParcial->setVisibilidadeComboAtributos(false
                ); //é escondido pois esta janela ja tem uma combo com os
                atributos
122         _widgetCategoriasMediaParcial = new WidgetCategoriasQual(atributosQual,
                 ui.stackedWidgetIntervalosMediaParcial->widget(1));
123         _widgetCategoriasMediaParcial->setVisibilidadeComboAtributos(false); //
                é escondido pois esta janela ja tem uma combo com os atributos
124         ui.stackedWidgetIntervalosMediaParcial->widget(0)->layout()->addWidget(
                _widgetIntervalosQuantMediaParcial);
125         ui.stackedWidgetIntervalosMediaParcial->widget(1)->layout()->addWidget(
                _widgetCategoriasMediaParcial);
126     }
127
128     void DialogSubstQuant::setupRLS(const vector<string> &atributosRLS, const
            vector<vector<size_t>> &indicesValidosRLS)
129     {
130         _widgetRegressaoLinearSimples = new WidgetRegressaoLinearSimples(
                _atributo, atributosRLS, indicesValidosRLS, this);
131         ui.groupBoxRLS->layout()->addWidget(_widgetRegressaoLinearSimples);
132     }
133
134     void DialogSubstQuant::setupEscolherDP(const vector<size_t> &
            indicesValidosDP)
135     {
136         _widgetEscolherDP = new WidgetEscolherDP(_atributo, indicesValidosDP,
                WidgetEscolherDP::Opcao::NADA, this);
137         ui.groupBoxDP->layout()->addWidget(_widgetEscolherDP);
138     }
139
140     void DialogSubstQuant::substituaComValor()
141     {
142         if (ui.radioButtonMediaParcial->isChecked() && !
                calcularMediaParcialClique())
143         {
144             return;
145         }
146
147         bool sucesso = false;
148
149         const string novoValorStr = ui.doubleSpinBoxValor->text().toLatin1().
                data();
150
151         ValorBase *novoValor = TabelaDeRegistros::obterInstancia()->gerarValor(
                _atributo, novoValorStr);
152
153         if (ui.radioButtonMissing->isChecked())
154         {
155             emit substituaMissings(_atributo, novoValor);
156             sucesso = true;
157         }
158         else if (ui.radioButtonTodos->isChecked())
159         {
160             emit substituaTodos(_atributo, novoValor);
161             sucesso = true;
162         }
163         else if (ui.radioButtonInvalidos->isChecked())
164         {
165             emit substituaInvalidos(_atributo, novoValor);
166             sucesso = true;
167         }
168         else if (ui.radioButtonIntervaloParaSubs->isChecked())
```

```
169     {
170         const bool missingsTmb = _widgetIntervalosQuantParaSubstituir->
                considerarMissings();
171         const bool invalidosTmb = _widgetIntervalosQuantParaSubstituir->
                considerarInvalidos();
172 
173         vector<TabelaDeRegistros::Intervalo> intervalos = vector<
                TabelaDeRegistros::Intervalo>();
174         _widgetIntervalosQuantParaSubstituir->gerarIntervalos(intervalos);
175 
176         emit substituaIntervalos(_atributo, novoValor, intervalos,
                missingsTmb, invalidosTmb);
177         sucesso = true;
178     }
179     else if (ui.radioButtonDestacados->isChecked())
180     {
181         emit substituaDestacados(_atributo, novoValor);
182         sucesso = true;
183     }
184 
185     delete novoValor;
186 
187     if (sucesso)
188     {
189         accept();
190     }
191 }
192 
193 void DialogSubstQuant::substituaComDP()
194 {
195     QMessageBox msgQuestion(QMessageBox::Question, "Aviso", QString::
            fromLatin1(
196         "Você selecionou o modo de substituição por uma distribuição de
                probabilidade.\n"
197         "Os valores serão gerados aleatóriamente de acordo com a
                distribuicao ativa.\n"
198         "Deseja continuar mesmo assim?"), QMessageBox::Yes|QMessageBox::No,
                this);
199     msgQuestion.button(QMessageBox::Yes)->setText(QString::fromLatin1("Sim"
            ));
200     msgQuestion.button(QMessageBox::No)->setText(QString::fromLatin1("Não")
            );
201 
202     const Distribuicao &distribuicao = _widgetEscolherDP->distribuicaoAtual
            ();
203 
204     if (ui.radioButtonMissing->isChecked())
205     {
206         emit substituaMissingsDP(_atributo, distribuicao);
207     }
208     else if (ui.radioButtonTodos->isChecked())
209     {
210         emit substituaTodosDP(_atributo, distribuicao);
211     }
212     else if (ui.radioButtonInvalidos->isChecked())
213     {
214         emit substituaInvalidosDP(_atributo, distribuicao);
215     }
216     else if (ui.radioButtonIntervaloParaSubs->isChecked())
217     {
218         const bool missingsTmb = _widgetIntervalosQuantParaSubstituir->
                considerarMissings();
219         const bool invalidosTmb = _widgetIntervalosQuantParaSubstituir->
                considerarInvalidos();
220 
221         vector<TabelaDeRegistros::Intervalo> intervalos = vector<
                TabelaDeRegistros::Intervalo>();
222         _widgetIntervalosQuantParaSubstituir->gerarIntervalos(intervalos);
223 
224         emit substituaIntervaloDP(_atributo, distribuicao, intervalos,
                missingsTmb, invalidosTmb);
225     }
226     else if (ui.radioButtonDestacados->isChecked())
227     {
228         emit substituaDestacadosDP(_atributo, distribuicao);
229     }
230 
231     accept();
232     return;
233 }
234 
235 void DialogSubstQuant::substituaComRLS()
```

```
236 {
237     const RegressaoLinearSimples &regressao = _widgetRegressaoLinearSimples
            ->regressaoLinearSimples();
238     const string atributoInd = _widgetRegressaoLinearSimples->atributoIndep
            ();
239
240     if (ui.radioButtonMissing->isChecked())
241     {
242         emit substituaMissingsRLS(_atributo, atributoInd, regressao);
243     }
244     else if (ui.radioButtonTodos->isChecked())
245     {
246         emit substituaTodosRLS(_atributo, atributoInd, regressao);
247     }
248     else if (ui.radioButtonInvalidos->isChecked())
249     {
250         emit substituaInvalidosRLS(_atributo, atributoInd, regressao);
251     }
252     else if (ui.radioButtonIntervaloParaSubs->isChecked())
253     {
254         const bool missingsTmb = _widgetIntervalosQuantParaSubstituir->
                considerarMissings();
255         const bool invalidosTmb = _widgetIntervalosQuantParaSubstituir->
                considerarInvalidos();
256
257         vector<TabelaDeRegistros::Intervalo> intervalos = vector<
                TabelaDeRegistros::Intervalo>();
258         _widgetIntervalosQuantParaSubstituir->gerarIntervalos(intervalos);
259
260         emit substituaIntervaloRLS(_atributo, atributoInd, regressao,
                intervalos, missingsTmb, invalidosTmb);
261     }
262     else if (ui.radioButtonDestacados->isChecked())
263     {
264         emit substituaDestacadosRLS(_atributo, atributoInd, regressao);
265     }
266
267     accept();
268     return;
269 }
270
271 void DialogSubstQuant::subtituirClique()
272 {
273     if (ui.radioButtonDP->isChecked())
274     {
275         substituaComDP();
276     }
277     else if (ui.radioButtonRLS->isChecked())
278     {
279         substituaComRLS();
280     }
281     else
282     {
283         substituaComValor();
284     }
285 }
286
287 void DialogSubstQuant::currentIndexChangedComboBoxAtributoMedParcial()
288 {
289     const string atributoSelecioando = ui.comboBoxAtributoMedParcial->
            currentText().toLatin1().data();
290
291     if (TabelaDeRegistros::obterInstancia()->ehQuant(atributoSelecioando))
292     {
293         _widgetIntervalosQuantMediaParcial->
                setAtributoAtualComboBoxAtributos(atributoSelecioando);
294         ui.stackedWidgetIntervalosMediaParcial->setCurrentWidget(ui.
                pageQuant);
295     }
296     else
297     {
298         _widgetCategoriasMediaParcial->setAtributoAtualComboBoxAtributos(
                atributoSelecioando);
299         ui.stackedWidgetIntervalosMediaParcial->setCurrentWidget(ui.
                pageQuali);
300     }
301 }
302
303 bool DialogSubstQuant::calcularMediaParcialClique()
304 {
305     const string atributoMediaParcial = ui.comboBoxAtributoMedParcial->
            currentText().toLatin1().data();
```

```
306
307     double media = numeric_limits<double>::quiet_NaN();
308
309     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(
            atributoMediaParcial))
310     {
311         vector<ValorBase *const> selecionados = vector<ValorBase *const>();
312         _widgetCategoriasMediaParcial->obterSelecionados(selecionados);
313
314         media = ColetorEstatisticas::obterInstancia()->
                calcMediaParcialBaseadoValores(_atributo, atributoMediaParcial
                , selecionados);
315
316         for (vector<ValorBase *const>::const_iterator citSelecionado =
                selecionados.cbegin();
317             citSelecionado != selecionados.cend();
318             citSelecionado++)
319         {
320             delete *citSelecionado;
321         }
322     }
323     else
324     {
325         vector<TabelaDeRegistros::Intervalo> intervalos = vector<
                TabelaDeRegistros::Intervalo>();
326
327         _widgetIntervalosQuantMediaParcial->gerarIntervalos(intervalos);
328         media = ColetorEstatisticas::obterInstancia()->
                calcMediaParcialBaseadoIntervalos(_atributo,
                atributoMediaParcial, intervalos);
329     }
330
331     if (isnan(media))
332     {
333         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Não foram encontrador valores para cálculo da
                média móvel!"));
334         return false;
335     }
336     else
337     {
338         ui.doubleSpinBoxValor->setValue(media);
339     }
340
341     return true;
342 }
```

Listagem B.62: dialog_subst_quant.cpp

```
1  #ifndef DIALOG_SUBST_QUANT_H
2  #define DIALOG_SUBST_QUANT_H
3
4  #include "ui_dialog_subst_quant.h"
5
6  #include "widget_categorias_qual.h"
7  #include "widget_intervalos_quant.h"
8  #include "widget_escolher_dp.h"
9  #include "widget_regressao_linear_simples.h"
10
11 using namespace std;
12
13 class DialogSubstQuant : public QDialog
14 {
15     Q_OBJECT
16
17 public:
18     DialogSubstQuant(
19         const string &atributo,
20         const vector<string> &atributosQuant,
21         const vector<vector<size_t>> &indicesValidosPorAtribQuant,
22         QWidget *parent = 0);
23     ~DialogSubstQuant();
24
25     void somenteDestacados(const bool ok);
26
27 private:
28     Ui::Analise_EdicaoDadosSubst ui;
29     WidgetCategoriasQual *_widgetCategoriasMediaParcial;
30     WidgetIntervalosQuant *_widgetIntervalosQuantMediaParcial;
```

```
31      WidgetIntervalosQuant *_widgetIntervalosQuantParaSubstituir;
32      WidgetRegressaoLinearSimples *_widgetRegressaoLinearSimples;
33      WidgetEscolherDP *_widgetEscolherDP;
34      string _atributo;
35      double _media;
36
37      void setupIntervalosLocalizar();
38      void setupMediaParcial();
39      void setupRLS(const vector<string> &atributosRLS, const vector<vector<
            size_t>> &indicesValidosRLS);
40      void setupEscolherDP(const vector<size_t> &indicesValidosDP);
41
42      void substituaComRLS();
43      void substituaComDP();
44      void substituaComValor();
45
46  private slots:
47      void subtituirClique();
48      void radioButtonAlterado();
49      void currentIndexChangedComboBoxAtributoMedParcial();
50      bool calcularMediaParcialClique();
51
52
53  signals:
54      void substituaMissings(const string&, ValorBase *const);
55      void substituaTodos(const string&, ValorBase *const);
56      void substituaInvalidos(const string&, ValorBase *const);
57      void substituaIntervalos(const string&, ValorBase *const, const vector<
            TabelaDeRegistros::Intervalo>&, const bool, const bool);
58      void substituaDestacados(const string&, ValorBase *const);
59
60      void substituaMissingsDP(const string&, const Distribuicao&);
61      void substituaTodosDP(const string&, const Distribuicao&);
62      void substituaInvalidosDP(const string&, const Distribuicao&);
63      void substituaIntervaloDP(const string&, const Distribuicao&, const
            vector<TabelaDeRegistros::Intervalo>&, const bool, const bool);
64      void substituaDestacadosDP(const string&, const Distribuicao&);
65
66      void substituaMissingsRLS(const string&, const string&, const
            RegressaoLinearSimples&);
67      void substituaTodosRLS(const string&, const string&, const
            RegressaoLinearSimples&);
68      void substituaInvalidosRLS(const string&, const string&, const
            RegressaoLinearSimples&);
69      void substituaIntervaloRLS(const string&, const string&, const
            RegressaoLinearSimples&, const vector<TabelaDeRegistros::Intervalo
            >&, const bool, const bool);
70      void substituaDestacadosRLS(const string&, const string&, const
            RegressaoLinearSimples&);
71  };
72
73  #endif // DIALOG_SUBST_QUANT_H
```

Listagem B.63: dialog_subst_quant.h

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <ui version="4.0">
3    <class>Analise_EdicaoDadosSubst</class>
4    <widget class="QDialog" name="Analise_EdicaoDadosSubst">
5     <property name="enabled">
6      <bool>true</bool>
7     </property>
8     <property name="geometry">
9      <rect>
10      <x>0</x>
11      <y>0</y>
12      <width>572</width>
13      <height>313</height>
14     </rect>
15    </property>
16    <property name="sizePolicy">
17     <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
18      <horstretch>0</horstretch>
19      <verstretch>0</verstretch>
20     </sizepolicy>
21    </property>
22    <property name="minimumSize">
23     <size>
24      <width>0</width>
```

```
25      <height>0</height>
26     </size>
27    </property>
28    <property name="maximumSize">
29     <size>
30      <width>16777215</width>
31      <height>16777215</height>
32     </size>
33    </property>
34    <property name="windowTitle">
35     <string>Substituir Quantitativo</string>
36    </property>
37    <property name="windowOpacity">
38     <double>1.000000000000000</double>
39    </property>
40    <property name="sizeGripEnabled">
41     <bool>false</bool>
42    </property>
43    <property name="modal">
44     <bool>false</bool>
45    </property>
46    <layout class="QVBoxLayout" name="verticalLayout_6">
47     <item>
48      <layout class="QFormLayout" name="formLayout">
49       <item row="0" column="0">
50        <widget class="QLabel" name="label">
51         <property name="text">
52          <string>Atributo:</string>
53         </property>
54        </widget>
55       </item>
56       <item row="0" column="1">
57        <widget class="QLabel" name="labelAtributo">
58         <property name="text">
59          <string>TextLabel</string>
60         </property>
61        </widget>
62       </item>
63      </layout>
64     </item>
65     <item>
66      <widget class="QSplitter" name="splitter">
67       <property name="orientation">
68        <enum>Qt::Vertical</enum>
69       </property>
70       <property name="childrenCollapsible">
71        <bool>false</bool>
72       </property>
73       <widget class="QGroupBox" name="groupBox">
74        <property name="sizePolicy">
75         <sizepolicy hsizetype="Preferred" vsizetype="Minimum">
76          <horstretch>0</horstretch>
77          <verstretch>0</verstretch>
78         </sizepolicy>
79        </property>
80        <property name="minimumSize">
81         <size>
82          <width>0</width>
83          <height>0</height>
84         </size>
85        </property>
86        <property name="title">
87         <string>Substituir os valores:</string>
88        </property>
89        <layout class="QVBoxLayout" name="verticalLayout_8">
90         <item>
91          <layout class="QHBoxLayout" name="horizontalLayout_2">
92           <property name="sizeConstraint">
93            <enum>QLayout::SetMinimumSize</enum>
94           </property>
95           <item>
96            <widget class="QRadioButton" name="radioButtonInvalidos">
97             <property name="text">
98              <string>Inválidos</string>
99             </property>
100            <property name="checked">
101             <bool>true</bool>
102            </property>
103           </widget>
104          </item>
105          <item>
106           <widget class="QRadioButton" name="radioButtonMissing">
```

```
107             <property name="text">
108              <string>Missing</string>
109             </property>
110            </widget>
111           </item>
112           <item>
113            <widget class="QRadioButton" name="radioButtonTodos">
114             <property name="text">
115              <string>Todos</string>
116             </property>
117            </widget>
118           </item>
119           <item>
120            <widget class="QRadioButton" name="radioButtonIntervaloParaSubs">
121             <property name="text">
122              <string>Intervalo</string>
123             </property>
124            </widget>
125           </item>
126           <item>
127            <widget class="QRadioButton" name="radioButtonDestacados">
128             <property name="enabled">
129              <bool>true</bool>
130             </property>
131             <property name="text">
132              <string>Destacados</string>
133             </property>
134            </widget>
135           </item>
136          </layout>
137         </item>
138         <item>
139          <widget class="QGroupBox" name="groupBoxIntervalosSubst">
140           <property name="sizePolicy">
141            <sizepolicy hsizetype="Preferred" vsizetype="Minimum">
142             <horstretch>0</horstretch>
143             <verstretch>0</verstretch>
144            </sizepolicy>
145           </property>
146           <property name="title">
147            <string>Intevalos</string>
148           </property>
149           <layout class="QVBoxLayout" name="verticalLayout_7"/>
150          </widget>
151         </item>
152        </layout>
153       </widget>
154       <widget class="QGroupBox" name="groupBox_2">
155        <property name="sizePolicy">
156         <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
157          <horstretch>0</horstretch>
158          <verstretch>0</verstretch>
159         </sizepolicy>
160        </property>
161        <property name="title">
162         <string>Substituir por:</string>
163        </property>
164        <layout class="QVBoxLayout" name="verticalLayout_5">
165         <item>
166          <layout class="QHBoxLayout" name="horizontalLayout_5">
167           <item>
168            <widget class="QRadioButton" name="radioButtoZero">
169             <property name="text">
170              <string>Zero</string>
171             </property>
172             <property name="checked">
173              <bool>true</bool>
174             </property>
175            </widget>
176           </item>
177           <item>
178            <widget class="QRadioButton" name="radioButtonMedia">
179             <property name="text">
180              <string>Média geral</string>
181             </property>
182             <property name="checked">
183              <bool>false</bool>
184             </property>
185            </widget>
186           </item>
187           <item>
188            <widget class="QRadioButton" name="radioButtonMediaParcial">
```

```
189             <property name="text">
190              <string>Média parcial</string>
191             </property>
192            </widget>
193           </item>
194           <item>
195            <widget class="QRadioButton" name="radioButtonDP">
196             <property name="text">
197              <string>Distribuição de probabilidade</string>
198             </property>
199            </widget>
200           </item>
201           <item>
202            <widget class="QRadioButton" name="radioButtonRLS">
203             <property name="text">
204              <string>Regressão linear simples</string>
205             </property>
206            </widget>
207           </item>
208          </layout>
209         </item>
210         <item>
211          <widget class="QStackedWidget" name="stackedWidgetMediaParcialDpRLS
                ">
212           <property name="currentIndex">
213            <number>0</number>
214           </property>
215           <widget class="QWidget" name="pageMediaParcial">
216            <layout class="QVBoxLayout" name="verticalLayout_3">
217             <property name="leftMargin">
218              <number>0</number>
219             </property>
220             <property name="topMargin">
221              <number>0</number>
222             </property>
223             <property name="rightMargin">
224              <number>0</number>
225             </property>
226             <property name="bottomMargin">
227              <number>0</number>
228             </property>
229             <item>
230              <widget class="QGroupBox" name="groupBox_3">
231               <property name="title">
232                <string>Média móvel</string>
233               </property>
234               <layout class="QVBoxLayout" name="verticalLayout_9">
235                <item>
236                 <layout class="QHBoxLayout" name="horizontalLayout">
237                  <item>
238                   <widget class="QLabel" name="labelFiltro">
239                    <property name="text">
240                     <string>Atributo Filtro:</string>
241                    </property>
242                   </widget>
243                  </item>
244                  <item>
245                   <widget class="QComboBox" name="comboBoxAtributoMedParcial
                        ">
246                    <property name="sizePolicy">
247                     <sizepolicy hsizetype="Expanding" vsizetype="Fixed">
248                      <horstretch>0</horstretch>
249                      <verstretch>0</verstretch>
250                     </sizepolicy>
251                    </property>
252                   </widget>
253                  </item>
254                  <item>
255                   <spacer name="horizontalSpacer_2">
256                    <property name="orientation">
257                     <enum>Qt::Horizontal</enum>
258                    </property>
259                    <property name="sizeHint" stdset="0">
260                     <size>
261                      <width>40</width>
262                      <height>20</height>
263                     </size>
264                    </property>
265                   </spacer>
266                  </item>
267                  <item>
268                   <widget class="QPushButton" name="
```

```
                    pushButtonCalcMediaParcial">
269                  <property name="text">
270                   <string>Calcular Média Parcial</string>
271                  </property>
272                 </widget>
273                </item>
274               </layout>
275              </item>
276              <item>
277               <widget class="QStackedWidget" name="
                      stackedWidgetIntervalosMediaParcial">
278                <property name="enabled">
279                 <bool>true</bool>
280                </property>
281                <property name="sizePolicy">
282                 <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
283                  <horstretch>0</horstretch>
284                  <verstretch>0</verstretch>
285                 </sizepolicy>
286                </property>
287                <property name="currentIndex">
288                 <number>1</number>
289                </property>
290                <widget class="QWidget" name="pageQuant">
291                 <layout class="QVBoxLayout" name="verticalLayout_2">
292                  <property name="leftMargin">
293                   <number>0</number>
294                  </property>
295                  <property name="topMargin">
296                   <number>0</number>
297                  </property>
298                  <property name="rightMargin">
299                   <number>0</number>
300                  </property>
301                  <property name="bottomMargin">
302                   <number>0</number>
303                  </property>
304                 </layout>
305                </widget>
306                <widget class="QWidget" name="pageQuali">
307                 <layout class="QVBoxLayout" name="verticalLayout">
308                  <property name="leftMargin">
309                   <number>0</number>
310                  </property>
311                  <property name="topMargin">
312                   <number>0</number>
313                  </property>
314                  <property name="rightMargin">
315                   <number>0</number>
316                  </property>
317                  <property name="bottomMargin">
318                   <number>0</number>
319                  </property>
320                 </layout>
321                </widget>
322               </widget>
323              </item>
324             </layout>
325            </widget>
326           </item>
327          </layout>
328         </widget>
329         <widget class="QWidget" name="pageDP">
330          <layout class="QVBoxLayout" name="verticalLayout_4">
331           <property name="leftMargin">
332            <number>0</number>
333           </property>
334           <property name="topMargin">
335            <number>0</number>
336           </property>
337           <property name="rightMargin">
338            <number>0</number>
339           </property>
340           <property name="bottomMargin">
341            <number>0</number>
342           </property>
343           <item>
344            <widget class="QGroupBox" name="groupBoxDP">
345             <property name="maximumSize">
346              <size>
347               <width>16777215</width>
348               <height>16777215</height>
```

```
349                 </size>
350                </property>
351                <property name="title">
352                 <string>Distribuição de Probabilidade</string>
353                </property>
354                <layout class="QVBoxLayout" name="verticalLayout_10"/>
355               </widget>
356              </item>
357             </layout>
358            </widget>
359            <widget class="QWidget" name="pageRLS">
360             <layout class="QVBoxLayout" name="verticalLayout_11">
361              <property name="leftMargin">
362               <number>0</number>
363              </property>
364              <property name="topMargin">
365               <number>0</number>
366              </property>
367              <property name="rightMargin">
368               <number>0</number>
369              </property>
370              <property name="bottomMargin">
371               <number>0</number>
372              </property>
373              <item>
374               <widget class="QGroupBox" name="groupBoxRLS">
375                <property name="title">
376                 <string>Regressão linear simples</string>
377                </property>
378                <layout class="QVBoxLayout" name="verticalLayout_12"/>
379               </widget>
380              </item>
381             </layout>
382            </widget>
383           </widget>
384          </item>
385          <item>
386           <layout class="QHBoxLayout" name="horizontalLayout_4">
387            <item>
388             <widget class="QLabel" name="label_4">
389              <property name="text">
390               <string>Valor:</string>
391              </property>
392             </widget>
393            </item>
394            <item>
395             <widget class="QDoubleSpinBox" name="doubleSpinBoxValor">
396              <property name="enabled">
397               <bool>true</bool>
398              </property>
399              <property name="sizePolicy">
400               <sizepolicy hsizetype="Expanding" vsizetype="Fixed">
401                <horstretch>0</horstretch>
402                <verstretch>0</verstretch>
403               </sizepolicy>
404              </property>
405              <property name="alignment">
406               <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
407              </property>
408              <property name="maximum">
409               <double>999999999.000000000000000</double>
410              </property>
411              <property name="singleStep">
412               <double>999999999.000000000000000</double>
413              </property>
414             </widget>
415            </item>
416           </layout>
417          </item>
418         </layout>
419        </widget>
420       </widget>
421      </item>
422      <item>
423       <layout class="QHBoxLayout" name="horizontalLayout_3">
424        <item>
425         <spacer name="horizontalSpacer">
426          <property name="orientation">
427           <enum>Qt::Horizontal</enum>
428          </property>
429          <property name="sizeHint" stdset="0">
430           <size>
```

```
431             <width>40</width>
432             <height>20</height>
433            </size>
434           </property>
435          </spacer>
436         </item>
437         <item>
438          <widget class="QPushButton" name="pushButtonSubstituir">
439           <property name="text">
440            <string>Substituir</string>
441           </property>
442          </widget>
443         </item>
444         <item>
445          <widget class="QPushButton" name="pushButtonCancelar">
446           <property name="text">
447            <string>Cancelar</string>
448           </property>
449          </widget>
450         </item>
451        </layout>
452       </item>
453      </layout>
454     </widget>
455     <layoutdefault spacing="6" margin="11"/>
456     <resources/>
457     <connections/>
458    </ui>
```

Listagem B.64: dialog_subst_quant.ui

```
1  #include "dialog_visualizar_graficos.h"
2
3  #include "gerador_graficos.h"
4  #include "coletor_estatisticas.h"
5  #include "escala_nomes.h"
6  #include "populador.h"
7
8  #include <cassert>
9
10 DialogVisualizarGraficos::DialogVisualizarGraficos(QWidget *parent)
11     : QDialog(parent)
12 {
13     ui.setupUi(this);
14
15     populaComboBoxXYCor();
16
17     setWindowFlags((windowFlags() | Qt::WindowMaximizeButtonHint) & ~Qt::
           WindowContextHelpButtonHint);
18
19     inicializarTabDiagramaDispersao2v();
20     inicializarTabDiagramaDispersao3v();
21     inicializarTabBarrasEmpilhadas();
22     inicializarTabDiagramaCaixas();
23 }
24
25
26 void DialogVisualizarGraficos::inicializarTabDiagramaDispersao2v()
27 {
28     _diagramaDispersao2v = new DiagramaDispersao(false, ui.
           groupBoxDiagramaDispersao2V); //deletado em ~
           DialogVisualizarGraficos
29     ui.groupBoxDiagramaDispersao2V->layout()->addWidget(
           _diagramaDispersao2v);
30
31     ui.comboBoxX2v->setCurrentText("Litologia");
32     ui.comboBoxY2v->setCurrentText("PROF");
33
34     connect(ui.comboBoxX2v, SIGNAL(currentIndexChanged(const QString&)),
           this, SLOT(currentIndexChangedComboBoxXY2v()));
35     connect(ui.comboBoxY2v, SIGNAL(currentIndexChanged(const QString&)),
           this, SLOT(currentIndexChangedComboBoxXY2v()));
36
37     currentIndexChangedComboBoxXY2v();
38 }
39
40 void DialogVisualizarGraficos::inicializarTabDiagramaDispersao3v()
41 {
```

```
42      _diagramaDispersao3v = new DiagramaDispersao(true, ui.
            groupBoxDiagramaDispersao3V); //deletado em ~
            DialogVisualizarGraficos
43      ui.groupBoxDiagramaDispersao3V->layout()->addWidget(
            _diagramaDispersao3v);
44
45      ui.comboBoxX3v->setCurrentText("Litologia");
46      ui.comboBoxY3v->setCurrentText("PROF");
47      ui.comboBoxCor3v->setCurrentIndex(ui.comboBoxCor3v->count() - 1);
48
49      connect(ui.comboBoxX3v, SIGNAL(currentIndexChanged(const QString&)),
            this, SLOT(currentIndexChangedComboBoxXY3v()));
50      connect(ui.comboBoxY3v, SIGNAL(currentIndexChanged(const QString&)),
            this, SLOT(currentIndexChangedComboBoxXY3v()));
51      connect(ui.comboBoxCor3v, SIGNAL(currentIndexChanged(const QString&)),
            this, SLOT(currentIndexChangedComboBoxXY3v()));
52
53      currentIndexChangedComboBoxXY3v();
54  }
55
56  void DialogVisualizarGraficos::inicializarTabBarrasEmpilhadas()
57  {
58      _multiBarras = new GraficoMultiBarras(ui.groupBoxBarras); //deletado em
            ~DialogVisualizarGraficos
59      ui.groupBoxBarras->layout()->addWidget(_multiBarras);
60
61      ui.comboBoxXBarras->setCurrentText("Litologia");
62      ui.comboBoxCorBarras->setCurrentText("PROF");
63
64      connect(ui.comboBoxXBarras, SIGNAL(currentIndexChanged(const QString&))
            , this, SLOT(currentIndexChangedComboBoxBarrasXCor()));
65      connect(ui.comboBoxCorBarras, SIGNAL(currentIndexChanged(const QString
            &)), this, SLOT(currentIndexChangedComboBoxBarrasXCor()));
66
67      currentIndexChangedComboBoxBarrasXCor();
68  }
69
70  void DialogVisualizarGraficos::inicializarTabDiagramaCaixas()
71  {
72      _diagramaCaixa = new DiagramaCaixa(ui.groupBoxDiagramaCaixa); //
            deletado em ~DialogVisualizarGraficos
73      ui.groupBoxDiagramaCaixa->layout()->addWidget(_diagramaCaixa);
74
75      connect(ui.comboBoxAtributoDiagramaCaixa, SIGNAL(currentIndexChanged(
            const QString&)), this, SLOT(
            currentIndexChangedComboBoxAtributoDiagramaCaixas()));
76
77      currentIndexChangedComboBoxAtributoDiagramaCaixas();
78  }
79
80
81  DialogVisualizarGraficos::~DialogVisualizarGraficos()
82  {
83      const void * address = static_cast<const void*>(&(*this));
84      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
85
86      delete _diagramaDispersao2v;
87      delete _diagramaDispersao3v;
88      delete _multiBarras;
89      delete _diagramaCaixa;
90  }
91
92  void DialogVisualizarGraficos::populaComboBoxXYCor() const
93  {
94      const vector<string> &atributos = TabelaDeRegistros::obterInstancia()->
            atributos();
95      const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
            ()->atributosQuantitativos();
96
97      Populador::populeComboBox(ui.comboBoxX2v, atributos);
98      Populador::populeComboBox(ui.comboBoxY2v, atributos);
99
100     Populador::populeComboBox(ui.comboBoxY3v, atributos);
101     Populador::populeComboBox(ui.comboBoxX3v, atributos);
102     Populador::populeComboBox(ui.comboBoxCor3v, atributos);
103
104     Populador::populeComboBox(ui.comboBoxXBarras, atributos);
105     Populador::populeComboBox(ui.comboBoxCorBarras, atributos);
106
107     Populador::populeComboBox(ui.comboBoxAtributoDiagramaCaixa,
            atributosQuant);
108
```

```
109     ui.comboBoxX2v->setCurrentIndex(0);
110     ui.comboBoxXBarras->setCurrentIndex(0);
111
112     if (atributos.size() > 1)
113     {
114         ui.comboBoxY2v->setCurrentIndex(atributos.size() - 1);
115         ui.comboBoxCorBarras->setCurrentIndex(atributos.size() - 1);
116     }
117     else
118     {
119         ui.comboBoxY2v->setCurrentIndex(0);
120         ui.comboBoxCorBarras->setCurrentIndex(0);
121     }
122 }
123
124 void DialogVisualizarGraficos::ploteDiagramaDispersao2v(const string &
        atribX, const string &atribY)
125 {
126     _diagramaDispersao2v->clearPoints();
127
128     //_diagramaDispersao2v->setUpdatesEnabled(false);
129     _diagramaDispersao2v->setAutoReplot(false);
130     _diagramaDispersao2v->setAxisAutoScale(false);
131     _diagramaDispersao2v->setAxisTitle(QwtPlot::xBottom, QString::
            fromLatin1(atribX.c_str()));
132     _diagramaDispersao2v->setAxisTitle(QwtPlot::yLeft, QString::fromLatin1(
            atribY.c_str()));
133
134     GeradorGraficos::Coordenadas coordenadas = GeradorGraficos::Coordenadas
            ();
135     GeradorGraficos::gerarDiagramaDispersao2v(atribX, atribY, coordenadas);
136
137     if (coordenadas.empty())
138     {
139         const string msg = "Não foi possível gerar o gráfico de dispersão
                com duas variáveis.\n"
140             "Um dos atributos \"" +atribX +"\" ou \"" +atribY
141             +"\" tem todos os valores inválidos!";
142
143         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1(msg.c_str()));
144         return;
145     }
146
147     for (GeradorGraficos::Coordenadas::const_iterator citCoordenada =
            coordenadas.cbegin();
148         citCoordenada != coordenadas.cend();
149         citCoordenada++)
150     {
151         const GeradorGraficos::Coordenada &coordenada = *citCoordenada;
152
153         _diagramaDispersao2v->appendPoint(QPointF(coordenada[0], coordenada
                [1]));
154     }
155
156     ajustarEixoDiagramaDispersao(_diagramaDispersao2v, atribX, QwtPlot::
            xBottom);
157     ajustarEixoDiagramaDispersao(_diagramaDispersao2v, atribY, QwtPlot::
            yLeft);
158
159     //_diagramaDispersao2v->setUpdatesEnabled(true);
160     _diagramaDispersao2v->setAutoReplot(true);
161     _diagramaDispersao2v->setAxisAutoScale(true);
162
163     _diagramaDispersao2v->replot();
164 }
165
166 void DialogVisualizarGraficos::ploteDiagramaDispersao3v(const string &
        atribX, const string &atribY, const string &atribCor)
167 {
168     vector<string> classesString = vector<string>();
169     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atribCor))
170     {
171         ColetorEstatisticas::obterInstancia()->todasCategoriasAtributoQual(
                atribCor, classesString);
172     }
173     else
174     {
175         classesString = obterIntervalosQuantitativosString(atribCor);
176     }
177
178     QList<QString> classesQString = converteToQListQString(classesString);
```

```
179
180     //_diagramaDispersao3v->setUpdatesEnabled(false);
181     _diagramaDispersao3v->setAutoReplot(false);
182     _diagramaDispersao3v->setAxisAutoScale(false);
183     _diagramaDispersao3v->setTotalCurvas(classesQString.size(),
            classesQString);
184     _diagramaDispersao3v->setAxisTitle(QwtPlot::xBottom, QString::
            fromLatin1(atribX.c_str()));
185     _diagramaDispersao3v->setAxisTitle(QwtPlot::yLeft, QString::fromLatin1(
            atribY.c_str()));
186     _diagramaDispersao3v->setTituloLegenda(QString::fromLatin1(atribCor.
            c_str()));
187
188     GeradorGraficos::Coordenadas coordenadas = GeradorGraficos::Coordenadas
            ();
189     GeradorGraficos::gerarDiagramaDispersao3v(atribX, atribY, atribCor,
            coordenadas);
190
191     if (coordenadas.empty())
192     {
193         const string msg = "Não foi possível gerar o gráfico de dispersão
                com três variáveis.\n"
194             "Um dos atributos \"" +atribX +"\", \"" +atribY +"\" ou \"" +
                    atribCor
195             +"\" tem todos os valores inválidos!";
196
197         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1(msg.c_str()));
198         return;
199     }
200
201     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atribCor))
202     {
203         for (GeradorGraficos::Coordenadas::const_iterator citCoord =
                coordenadas.cbegin();
204             citCoord != coordenadas.cend();
205             citCoord++)
206         {
207             const GeradorGraficos::Coordenada &coordenada = *citCoord;
208
209             _diagramaDispersao3v->appendPoint(QPointF(coordenada[0],
                    coordenada[1]), coordenada[2]);
210         }
211     }
212     else
213     {
214         for (GeradorGraficos::Coordenadas::const_iterator citCoord =
                coordenadas.cbegin();
215             citCoord != coordenadas.cend();
216             citCoord++)
217         {
218             const GeradorGraficos::Coordenada &coordenada = *citCoord;
219
220             _diagramaDispersao3v->appendPoint(QPointF(coordenada[0],
                    coordenada[1]), ColetorEstatisticas::obterInstancia()->
                    indiceClasseQuant(coordenada[2], atribCor));
221         }
222     }
223
224     ajustarEixoDiagramaDispersao(_diagramaDispersao3v, atribX, QwtPlot::
            xBottom);
225     ajustarEixoDiagramaDispersao(_diagramaDispersao3v, atribY, QwtPlot::
            yLeft);
226
227     //_diagramaDispersao3v->setUpdatesEnabled(true);
228     _diagramaDispersao3v->setAutoReplot(true);
229     _diagramaDispersao3v->setAxisAutoScale(true);
230
231     _diagramaDispersao3v->replot();
232 }
233
234 QList<QString> DialogVisualizarGraficos::converteToQListQString(const
        vector<string> &classes) const
235 {
236     QList<QString> lista;
237
238     for (vector<string>::const_iterator citClasse = classes.cbegin();
239         citClasse != classes.cend();
240         citClasse++)
241     {
242         lista.push_back(QString::fromLatin1(citClasse->c_str()));
243     }
```

```
244
245     return lista;
246 }
247
248 void DialogVisualizarGraficos::ajustarEixoDiagramaDispersao(
        DiagramaDispersao *diagramaDispersao, const string &atrib, const int
        axisId) const
249 {
250     diagramaDispersao->setAxisTitle(axisId, QString::fromLatin1(atrib.c_str
            ()));
251
252     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atrib))
253     {
254         vector<string> valQualitativos = vector<string>();
255         ColetorEstatisticas::obterInstancia()->todasCategoriasAtributoQual(
                atrib, valQualitativos);
256
257         const size_t max = valQualitativos.size();
258         diagramaDispersao->setAxisScale(axisId, -1, max, 1);
259
260         diagramaDispersao->setAxisScaleDraw(axisId, new EscalaNomes(
                valQualitativos)); //Quando seta uma escala é chamado a escala
                 antiga é excluida, Quando destroi o _diagramaDispersao2v tmb
                destroi a escala
261         diagramaDispersao->updateAxes();
262
263         if (axisId == QwtPlot::xBottom)
264         {
265             diagramaDispersao->setAxisLabelRotation(QwtPlot::xBottom, 45);
266             diagramaDispersao->setAxisLabelAlignment(QwtPlot::xBottom, Qt::
                    AlignRight);
267         }
268     }
269     else
270     {
271         const ColetorEstatisticas::EstatisticaQuant &estatisticaQuant =
                ColetorEstatisticas::obterInstancia()->
                estatisticaAtributoQuantitativo(atrib);
272         diagramaDispersao->setAxisScale(axisId, estatisticaQuant.minimo -
                1, estatisticaQuant.maximo + 1);
273
274         diagramaDispersao->setAxisScaleDraw(axisId, new EscalaNomes()); //
                Quando seta uma escala é chamado a escala antiga é excluida,
                Quando destroi o _diagramaDispersao2v tmb destroi a escala
275         diagramaDispersao->updateAxes();
276
277         if (axisId == QwtPlot::xBottom)
278         {
279             diagramaDispersao->setAxisLabelRotation(QwtPlot::xBottom, 0);
280             diagramaDispersao->setAxisLabelAlignment(QwtPlot::xBottom, Qt::
                    AlignCenter);
281         }
282     }
283 }
284
285 void DialogVisualizarGraficos::currentIndexChangedComboBoxXY2v()
286 {
287     const string atributoX = ui.comboBoxX2v->currentText().toLatin1().data
            ();
288     const string atributoY = ui.comboBoxY2v->currentText().toLatin1().data
            ();
289
290     ploteDiagramaDispersao2v(atributoX, atributoY);
291 }
292
293 void DialogVisualizarGraficos::currentIndexChangedComboBoxXY3v()
294 {
295     const string atributoX = ui.comboBoxX3v->currentText().toLatin1().data
            ();
296     const string atributoY = ui.comboBoxY3v->currentText().toLatin1().data
            ();
297     const string atributoCor = ui.comboBoxCor3v->currentText().toLatin1().
            data();
298
299     ploteDiagramaDispersao3v(atributoX, atributoY, atributoCor);
300 }
301
302 void DialogVisualizarGraficos::currentIndexChangedComboBoxBarrasXCor()
303 {
304     const string atributoX = ui.comboBoxXBarras->currentText().toLatin1().
            data();
305     const string atributoY = ui.comboBoxCorBarras->currentText().toLatin1()
```

```
            .data();
306
307     ploteMultiBarras(atributoX, atributoY);
308 }
309
310 void DialogVisualizarGraficos::
        currentIndexChangedComboBoxAtributoDiagramaCaixas()
311 {
312     const string atributo = ui.comboBoxAtributoDiagramaCaixa->currentText()
            .toLatin1().data();
313     ploteDiagramaCaixas(atributo);
314 }
315
316 void DialogVisualizarGraficos::ploteMultiBarras(const string &atribX, const
        string &atribCor)
317 {
318     //_multiBarras->setUpdatesEnabled(false);
319     _multiBarras->setAutoReplot(false);
320     _multiBarras->setAxisAutoScale(false);
321
322     vector<GeradorGraficos::Frequencias> frequencias = vector<
            GeradorGraficos::Frequencias>();
323     GeradorGraficos::gerarGraficoMultBarras(atribX, atribCor, frequencias);
324
325     if (frequencias.empty())
326     {
327         const string msg = "Não foi possível gerar o gráfico multibarras.\n
                "
328             "Um dos atributos \"" +atribX +"\" ou \"" +atribCor
329             +"\" tem todos os valores inválidos!";
330
331         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1(msg.c_str()));
332         return;
333     }
334
335     const size_t n = frequencias.size();
336     const size_t m = frequencias[0].size();
337
338     _multiBarras->populate(frequencias, obterTitulosBarras(atribCor));
339     _multiBarras->setAxisTitle(QwtPlot::yLeft, QString::fromLatin1("
            Frequência"));
340     _multiBarras->setAxisTitle(QwtPlot::xBottom, QString::fromLatin1(atribX
            .c_str()));
341     _multiBarras->setTituloLegenda(QString::fromLatin1(atribCor.c_str()));
342
343     ajustarEixoXBarrasEmpilhadas(atribX);
344
345     //_multiBarras->setUpdatesEnabled(true);
346     _multiBarras->setAutoReplot(true);
347     _multiBarras->setAxisAutoScale(true);
348
349     _multiBarras->replot();
350 }
351
352 void DialogVisualizarGraficos::ploteDiagramaCaixas(const string &atributo)
353 {
354     const ColetorEstatisticas::EstatisticaQuant &estatistica =
            ColetorEstatisticas::obterInstancia()->
            estatisticaAtributoQuantitativo(atributo);
355
356     _diagramaCaixa->setDados(
357         atributo,
358         estatistica.maximo,
359         estatistica.whiskerSuperiorDiagramaCaixa,
360         estatistica.quartilSuperior,
361         estatistica.mediana,
362         estatistica.quartilInferior,
363         estatistica.whiskerInferiorDiagramaCaixa,
364         estatistica.minimo,
365         estatistica.outliersDiagramaCaixa);
366 }
367
368 void DialogVisualizarGraficos::ajustarEixoXBarrasEmpilhadas(const string &
        atribX) const
369 {
370     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atribX))
371     {
372         vector<string> valQualitativos = vector<string>();
373         ColetorEstatisticas::obterInstancia()->todasCategoriasAtributoQual(
                atribX, valQualitativos);
374
```

```
375             const size_t max = valQualitativos.size();
376             _multiBarras->setAxisScale(QwtPlot::xBottom, -1, max, 1);
377 
378             _multiBarras->setAxisScaleDraw(QwtPlot::xBottom, new EscalaNomes(
                    valQualitativos)); //Quando seta uma escala é chamado a escala
                     antiga é excluida, Quando destroi o _diagramaDispersao2v tmb
                    destroi a escala
379         }
380         else
381         {
382             vector<string> &intervalosString =
                    obterIntervalosQuantitativosString(atribX);
383 
384             _multiBarras->setAxisScale(QwtPlot::xBottom, 0, intervalosString.
                    size(), 1);
385             _multiBarras->setAxisScaleDraw(QwtPlot::xBottom, new EscalaNomes(
                    intervalosString)); //Quando seta uma escala é chamado a
                    escala antiga é excluida, Quando destroi o
                    _diagramaDispersao2v tmb destroi a escala
386 
387         }
388 
389         _multiBarras->ajustarLabelEixo(QwtPlot::xBottom, Qt::AlignRight, 45);
390         _multiBarras->updateAxes();
391     }
392 
393     vector<string> DialogVisualizarGraficos::obterIntervalosQuantitativosString
            (const string &atrib) const
394     {
395         const ColetorEstatisticas::FrequenciasQuantContinuo &frequencias =
                ColetorEstatisticas::obterInstancia()->
                frequenciasAtributoQuantContinuo(atrib);
396 
397         vector<string> intervalosString = vector<string>();
398 
399         for (ColetorEstatisticas::FrequenciasQuantContinuo::const_iterator
                citFreq = frequencias.cbegin();
400             citFreq != frequencias.cend();
401             citFreq++)
402         {
403             const string inferior = Utilidades::dblToStdStr(citFreq->first.
                    first);
404 
405             const string superior = Utilidades::dblToStdStr(citFreq->first.
                    second);
406 
407             const string concatenacao = inferior + " |-- " + superior;
408 
409             intervalosString.push_back(concatenacao);
410         }
411 
412         return intervalosString;
413     }
414 
415     vector<string> DialogVisualizarGraficos::obterTitulosBarras(const string &
            atribY)
416     {
417         if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atribY))
418         {
419             vector<string> valQualitativos = vector<string>();
420             ColetorEstatisticas::obterInstancia()->todasCategoriasAtributoQual(
                    atribY, valQualitativos);
421             return valQualitativos;
422         }
423         return obterIntervalosQuantitativosString(atribY);
424     }
```

Listagem B.65: dialog_visualizar_graficos.cpp

```
1  #ifndef DIALOG_VISUALIZAR_GRAFICOS_H
2  #define DIALOG_VISUALIZAR_GRAFICOS_H
3 
4  #include "ui_dialog_visualizar_graficos.h"
5 
6  #include "diagrama_scatter.h"
7  #include "grafico_multibarras.h"
8  #include "diagrama_caixa.h"
9 
10 using namespace std;
```

```
11
12 class DialogVisualizarGraficos : public QDialog
13 {
14     Q_OBJECT
15
16 public:
17     DialogVisualizarGraficos(QWidget *parent = 0);
18     ~DialogVisualizarGraficos();
19
20 private:
21     Ui::DialogVisualizarGraficos ui;
22     DiagramaDispersao *_diagramaDispersao2v;
23     DiagramaDispersao *_diagramaDispersao3v;
24     GraficoMultiBarras *_multiBarras;
25     DiagramaCaixa *_diagramaCaixa;
26
27     void populaComboBoxXYCor() const;
28     void ploteDiagramaDispersao2v(const string &atribX, const string &
           atribY);
29     void ploteDiagramaDispersao3v(const string &atribX, const string &
           atribY, const string &atribCor);
30     void ploteMultiBarras(const string &atribX, const string &atribCor);
31     void ploteDiagramaCaixas(const string &atributo);
32     void inicializarTabDiagramaDispersao2v();
33     void inicializarTabDiagramaDispersao3v();
34     void inicializarTabBarrasEmpilhadas();
35     void inicializarTabDiagramaCaixas();
36     QList<QString> converteToQListQString(const vector<string> &classes)
           const;
37     void ajustarEixoDiagramaDispersao(DiagramaDispersao *diagramaDispersao,
            const string &atrib, const int axisId) const;
38     void ajustarEixo(const string &atrib, const int axisId) const;
39     void ajustarEixoXBarrasEmpilhadas(const string &atribX) const;
40     vector<string> obterTitulosBarras(const string &atribY);
41     vector<string> obterIntervalosQuantitativosString(const string &atrib)
           const;
42
43 private slots:
44     void currentIndexChangedComboBoxXY2v();
45     void currentIndexChangedComboBoxXY3v();
46     void currentIndexChangedComboBoxBarrasXCor();
47     void currentIndexChangedComboBoxAtributoDiagramaCaixas();
48 };
49
50 #endif // DIALOG_VISUALIZAR_GRAFICOS_H
```

Listagem B.66: dialog_visualizar_graficos.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>DialogVisualizarGraficos</class>
4   <widget class="QDialog" name="DialogVisualizarGraficos">
5    <property name="windowModality">
6     <enum>Qt::NonModal</enum>
7    </property>
8    <property name="geometry">
9     <rect>
10     <x>0</x>
11     <y>0</y>
12     <width>761</width>
13     <height>548</height>
14    </rect>
15   </property>
16   <property name="windowTitle">
17    <string>Visualização dos Dados</string>
18   </property>
19   <property name="sizeGripEnabled">
20    <bool>false</bool>
21   </property>
22   <layout class="QVBoxLayout" name="verticalLayout_2">
23    <item>
24     <widget class="QTabWidget" name="tabWidget">
25      <property name="maximumSize">
26       <size>
27        <width>16777215</width>
28        <height>16777215</height>
29       </size>
30      </property>
31      <property name="currentIndex">
```

```
32          <number>3</number>
33         </property>
34         <widget class="QWidget" name="tabDiagramaDispersao2V">
35          <attribute name="title">
36           <string>Diagrama Dispersão (2V)</string>
37          </attribute>
38          <layout class="QVBoxLayout" name="verticalLayout">
39           <item>
40            <widget class="QGroupBox" name="groupBoxEixos">
41             <property name="sizePolicy">
42              <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
43               <horstretch>0</horstretch>
44               <verstretch>0</verstretch>
45              </sizepolicy>
46             </property>
47             <property name="title">
48              <string>Eixos</string>
49             </property>
50             <layout class="QHBoxLayout" name="horizontalLayout">
51              <item>
52               <layout class="QFormLayout" name="formLayout_3">
53                <item row="0" column="0">
54                 <widget class="QLabel" name="labelX">
55                  <property name="minimumSize">
56                   <size>
57                    <width>25</width>
58                    <height>0</height>
59                   </size>
60                  </property>
61                  <property name="text">
62                   <string>X:</string>
63                  </property>
64                 </widget>
65                </item>
66                <item row="0" column="1">
67                 <widget class="QComboBox" name="comboBoxX2v">
68                  <property name="sizePolicy">
69                   <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
70                    <horstretch>0</horstretch>
71                    <verstretch>0</verstretch>
72                   </sizepolicy>
73                  </property>
74                  <property name="minimumSize">
75                   <size>
76                    <width>200</width>
77                    <height>0</height>
78                   </size>
79                  </property>
80                  <property name="maximumSize">
81                   <size>
82                    <width>200</width>
83                    <height>16777215</height>
84                   </size>
85                  </property>
86                 </widget>
87                </item>
88                <item row="1" column="0">
89                 <widget class="QLabel" name="labelY">
90                  <property name="sizePolicy">
91                   <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
92                    <horstretch>0</horstretch>
93                    <verstretch>0</verstretch>
94                   </sizepolicy>
95                  </property>
96                  <property name="minimumSize">
97                   <size>
98                    <width>25</width>
99                    <height>0</height>
100                  </size>
101                 </property>
102                 <property name="text">
103                  <string>Y:</string>
104                 </property>
105                </widget>
106               </item>
107               <item row="1" column="1">
108                <widget class="QComboBox" name="comboBoxY2v">
109                 <property name="sizePolicy">
110                  <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
111                   <horstretch>0</horstretch>
112                   <verstretch>0</verstretch>
113                  </sizepolicy>
```

```
114                 </property>
115                 <property name="minimumSize">
116                  <size>
117                   <width>200</width>
118                   <height>0</height>
119                  </size>
120                 </property>
121                 <property name="maximumSize">
122                  <size>
123                   <width>200</width>
124                   <height>16777215</height>
125                  </size>
126                 </property>
127                </widget>
128               </item>
129              </layout>
130             </item>
131             <item>
132              <spacer name="horizontalSpacer">
133               <property name="orientation">
134                <enum>Qt::Horizontal</enum>
135               </property>
136               <property name="sizeHint" stdset="0">
137                <size>
138                 <width>561</width>
139                 <height>20</height>
140                </size>
141               </property>
142              </spacer>
143             </item>
144            </layout>
145           </widget>
146          </item>
147          <item>
148           <widget class="QGroupBox" name="groupBoxDiagramaDispersao2V">
149            <property name="title">
150             <string>Diagrama Dispersão</string>
151            </property>
152            <layout class="QVBoxLayout" name="verticalLayout_3"/>
153           </widget>
154          </item>
155         </layout>
156        </widget>
157        <widget class="QWidget" name="tabDiagramaDispersao3V">
158         <attribute name="title">
159          <string>Diagrama Dispersão (3V)</string>
160         </attribute>
161         <layout class="QVBoxLayout" name="verticalLayout_12">
162          <item>
163           <widget class="QGroupBox" name="groupBoxEixos_3">
164            <property name="sizePolicy">
165             <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
166              <horstretch>0</horstretch>
167              <verstretch>0</verstretch>
168             </sizepolicy>
169            </property>
170            <property name="title">
171             <string>Eixos</string>
172            </property>
173            <layout class="QHBoxLayout" name="horizontalLayout_4">
174             <item>
175              <layout class="QFormLayout" name="formLayout">
176               <item row="0" column="0">
177                <widget class="QLabel" name="labelX_2">
178                 <property name="minimumSize">
179                  <size>
180                   <width>25</width>
181                   <height>0</height>
182                  </size>
183                 </property>
184                 <property name="text">
185                  <string>X:</string>
186                 </property>
187                </widget>
188               </item>
189               <item row="0" column="1">
190                <widget class="QComboBox" name="comboBoxX3v">
191                 <property name="sizePolicy">
192                  <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
193                   <horstretch>0</horstretch>
194                   <verstretch>0</verstretch>
195                  </sizepolicy>
```

```
196                 </property>
197                 <property name="minimumSize">
198                  <size>
199                   <width>200</width>
200                   <height>0</height>
201                  </size>
202                 </property>
203                 <property name="maximumSize">
204                  <size>
205                   <width>200</width>
206                   <height>16777215</height>
207                  </size>
208                 </property>
209                </widget>
210               </item>
211               <item row="1" column="0">
212                <widget class="QLabel" name="labelY_2">
213                 <property name="sizePolicy">
214                  <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
215                   <horstretch>0</horstretch>
216                   <verstretch>0</verstretch>
217                  </sizepolicy>
218                 </property>
219                 <property name="minimumSize">
220                  <size>
221                   <width>25</width>
222                   <height>0</height>
223                  </size>
224                 </property>
225                 <property name="text">
226                  <string>Y:</string>
227                 </property>
228                </widget>
229               </item>
230               <item row="1" column="1">
231                <widget class="QComboBox" name="comboBoxY3v">
232                 <property name="sizePolicy">
233                  <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
234                   <horstretch>0</horstretch>
235                   <verstretch>0</verstretch>
236                  </sizepolicy>
237                 </property>
238                 <property name="minimumSize">
239                  <size>
240                   <width>200</width>
241                   <height>0</height>
242                  </size>
243                 </property>
244                 <property name="maximumSize">
245                  <size>
246                   <width>200</width>
247                   <height>16777215</height>
248                  </size>
249                 </property>
250                </widget>
251               </item>
252               <item row="2" column="0">
253                <widget class="QLabel" name="label">
254                 <property name="text">
255                  <string>Cor:</string>
256                 </property>
257                </widget>
258               </item>
259               <item row="2" column="1">
260                <widget class="QComboBox" name="comboBoxCor3v">
261                 <property name="sizePolicy">
262                  <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
263                   <horstretch>0</horstretch>
264                   <verstretch>0</verstretch>
265                  </sizepolicy>
266                 </property>
267                 <property name="minimumSize">
268                  <size>
269                   <width>200</width>
270                   <height>0</height>
271                  </size>
272                 </property>
273                </widget>
274               </item>
275              </layout>
276             </item>
277             <item>
```

```
278             <spacer name="horizontalSpacer_3">
279              <property name="orientation">
280               <enum>Qt::Horizontal</enum>
281              </property>
282              <property name="sizeHint" stdset="0">
283               <size>
284                <width>561</width>
285                <height>20</height>
286               </size>
287              </property>
288             </spacer>
289            </item>
290           </layout>
291          </widget>
292         </item>
293         <item>
294          <widget class="QGroupBox" name="groupBoxDiagramaDispersao3V">
295           <property name="title">
296            <string>Diagrama Dispersão</string>
297           </property>
298           <layout class="QVBoxLayout" name="verticalLayout_15"/>
299          </widget>
300         </item>
301        </layout>
302       </widget>
303       <widget class="QWidget" name="tabBarras">
304        <attribute name="title">
305         <string>Barras Empilhadas</string>
306        </attribute>
307        <layout class="QVBoxLayout" name="verticalLayout_4">
308         <item>
309          <widget class="QGroupBox" name="groupBoxEixos_2">
310           <property name="sizePolicy">
311            <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
312             <horstretch>0</horstretch>
313             <verstretch>0</verstretch>
314            </sizepolicy>
315           </property>
316           <property name="title">
317            <string>Eixos</string>
318           </property>
319           <layout class="QHBoxLayout" name="horizontalLayout_2">
320            <item>
321             <layout class="QFormLayout" name="formLayout_2">
322              <item row="0" column="0">
323               <widget class="QLabel" name="labelXHist">
324                <property name="minimumSize">
325                 <size>
326                  <width>25</width>
327                  <height>0</height>
328                 </size>
329                </property>
330                <property name="text">
331                 <string>X:</string>
332                </property>
333               </widget>
334              </item>
335              <item row="0" column="1">
336               <widget class="QComboBox" name="comboBoxXBarras">
337                <property name="sizePolicy">
338                 <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
339                  <horstretch>0</horstretch>
340                  <verstretch>0</verstretch>
341                 </sizepolicy>
342                </property>
343                <property name="minimumSize">
344                 <size>
345                  <width>200</width>
346                  <height>0</height>
347                 </size>
348                </property>
349                <property name="maximumSize">
350                 <size>
351                  <width>200</width>
352                  <height>16777215</height>
353                 </size>
354                </property>
355               </widget>
356              </item>
357              <item row="1" column="0">
358               <widget class="QLabel" name="labelYHist">
359                <property name="sizePolicy">
```

```
360                 <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
361                  <horstretch>0</horstretch>
362                  <verstretch>0</verstretch>
363                 </sizepolicy>
364                </property>
365                <property name="minimumSize">
366                 <size>
367                  <width>25</width>
368                  <height>0</height>
369                 </size>
370                </property>
371                <property name="text">
372                 <string>Cor:</string>
373                </property>
374               </widget>
375              </item>
376              <item row="1" column="1">
377               <widget class="QComboBox" name="comboBoxCorBarras">
378                <property name="sizePolicy">
379                 <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
380                  <horstretch>0</horstretch>
381                  <verstretch>0</verstretch>
382                 </sizepolicy>
383                </property>
384                <property name="minimumSize">
385                 <size>
386                  <width>200</width>
387                  <height>0</height>
388                 </size>
389                </property>
390                <property name="maximumSize">
391                 <size>
392                  <width>200</width>
393                  <height>16777215</height>
394                 </size>
395                </property>
396               </widget>
397              </item>
398             </layout>
399            </item>
400            <item>
401             <spacer name="horizontalSpacer_2">
402              <property name="orientation">
403               <enum>Qt::Horizontal</enum>
404              </property>
405              <property name="sizeHint" stdset="0">
406               <size>
407                <width>561</width>
408                <height>20</height>
409               </size>
410              </property>
411             </spacer>
412            </item>
413           </layout>
414          </widget>
415         </item>
416         <item>
417          <widget class="QGroupBox" name="groupBoxBarras">
418           <property name="title">
419            <string>Gráfico de barras</string>
420           </property>
421           <layout class="QVBoxLayout" name="verticalLayout_11"/>
422          </widget>
423         </item>
424        </layout>
425       </widget>
426       <widget class="QWidget" name="tabDiagramaCaixa">
427        <attribute name="title">
428         <string>Diagrama de Caixas</string>
429        </attribute>
430        <layout class="QVBoxLayout" name="verticalLayout_6">
431         <item>
432          <layout class="QHBoxLayout" name="horizontalLayout_3">
433           <item>
434            <widget class="QLabel" name="labelAtributo">
435             <property name="text">
436              <string>Atributo:</string>
437             </property>
438            </widget>
439           </item>
440           <item>
441            <widget class="QComboBox" name="comboBoxAtributoDiagramaCaixa">
```

```
442             <property name="sizePolicy">
443              <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
444               <horstretch>0</horstretch>
445               <verstretch>0</verstretch>
446              </sizepolicy>
447             </property>
448             <property name="minimumSize">
449              <size>
450               <width>200</width>
451               <height>0</height>
452              </size>
453             </property>
454            </widget>
455           </item>
456           <item>
457            <spacer name="horizontalSpacer_4">
458             <property name="orientation">
459              <enum>Qt::Horizontal</enum>
460             </property>
461             <property name="sizeHint" stdset="0">
462              <size>
463               <width>40</width>
464               <height>20</height>
465              </size>
466             </property>
467            </spacer>
468           </item>
469          </layout>
470         </item>
471         <item>
472          <widget class="QGroupBox" name="groupBoxDiagramaCaixa">
473           <property name="sizePolicy">
474            <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
475             <horstretch>0</horstretch>
476             <verstretch>0</verstretch>
477            </sizepolicy>
478           </property>
479           <property name="title">
480            <string>Diagrama de Caixas</string>
481           </property>
482           <layout class="QVBoxLayout" name="verticalLayout_5"/>
483          </widget>
484         </item>
485        </layout>
486       </widget>
487      </widget>
488     </item>
489    </layout>
490   </widget>
491   <layoutdefault spacing="6" margin="11"/>
492   <resources/>
493   <connections/>
494  </ui>
```

Listagem B.67: dialog_visualizar_graficos.ui

```
1  #include "tela_inicial.h"
2
3  #include "widget_analise_exploracao_dados.h"
4
5  TelaInicial::TelaInicial(QWidget *parent)
6      : QMainWindow(parent)
7  {
8      ui.setupUi(this);
9
10     this->setCentralWidget(new WidgetAnaliseExploracaoDados());
11 }
12
13 TelaInicial::~TelaInicial()
14 {
15     const void * address = static_cast<const void*>(&(*this));
16     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
17 }
```

Listagem B.68: tela_inicial.cpp

```
#ifndef TELA_INICIAL_H
#define TELA_INICIAL_H

#include <QtWidgets/QMainWindow>
#include "ui_tela_inicial.h"

class TelaInicial : public QMainWindow
{
    Q_OBJECT

public:
    TelaInicial(QWidget *parent = 0);
    ~TelaInicial();

private:
    Ui::TelaInicialClass ui;
};

#endif // TELA_INICIAL_H
```

Listagem B.69: tela_inicial.h

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>TelaInicialClass</class>
 <widget class="QMainWindow" name="TelaInicialClass">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1024</width>
    <height>768</height>
   </rect>
  </property>
  <property name="minimumSize">
   <size>
    <width>0</width>
    <height>0</height>
   </size>
  </property>
  <property name="windowTitle">
   <string>Análise e Tratamento dos Dados</string>
  </property>
  <property name="windowIcon">
   <iconset resource="tela_inicial.qrc">
    <normaloff>:/AnaliseTratamento/Resources/icone.ico</normaloff>:/
        AnaliseTratamento/Resources/icone.ico</iconset>
  </property>
  <widget class="QWidget" name="centralWidget"/>
 </widget>
 <layoutdefault spacing="6" margin="11"/>
 <resources>
  <include location="tela_inicial.qrc"/>
 </resources>
 <connections/>
</ui>
```

Listagem B.70: tela_inicial.ui

```
<RCC>
    <qresource prefix="/AnaliseTratamento">
    <file>Resources/icone.ico</file>
    </qresource>
</RCC>
```

Listagem B.71: tela_inicial.qrc

```
#include "widget_analise_exploracao_dados.h"

#include "dialog_edicao_dados.h"
#include "dialog_visualizar_graficos.h"
#include "dialog_importacao_dados.h"

```

```
7  #include "populador.h"
8  #include "coletor_estatisticas.h"
9  #include "gerador_graficos.h"
10
11 #ifndef TCC_RODOLFO
12 #include "thread_worker_coletar_estatisticas.h"
13 #include "integracao_treinamento.h"
14 #include "escolha_poco.h"
15 #include "salvar_bd.h"
16 #include "dao_poco.h"
17 #include "thread_worker_salvarBD.h"
18 #endif
19
20 #include <fstream>
21
22 WidgetAnaliseExploracaoDados::WidgetAnaliseExploracaoDados(QWidget *parent)
       : WidgetProgresso(parent)
23 {
24     ui.setupUi(this);
25
26     this->_histograma = NULL;
27     this->_graficoBarras = NULL;
28
29     _primeiraInicializacao = true;
30
31     //nao pode editar tabela estatisticas
32     ui.tableWidgetEstatisticas->setEditTriggers(QAbstractItemView::
           NoEditTriggers);
33     ui.tableWidgetEstatisticas->verticalHeader()->setVisible(false);
34     ui.tableWidgetEstatisticas->horizontalHeader()->setVisible(false);
35     ui.tableWidgetEstatisticas->horizontalHeader()->setSectionResizeMode(
           QHeaderView::Stretch);
36
37 #ifndef TCC_RODOLFO
38     connect(ui.pushButtonCarregar, SIGNAL(clicked()), this, SLOT(
           carregarBDClick()));
39 #else
40     connect(ui.pushButtonCarregar, SIGNAL(clicked()), this, SLOT(
           carregarArquivoClick()));
41 #endif
42
43     habilitarEventos(false);
44
45     ui.widgetHistogramaGraficoBarras->setAutoFillBackground(true);
46     //QPalette pal = ui.widgetHistogramaGraficoBarras->palette();
47     //pal.setColor(QPalette::Window, Qt::white);
48     //ui.widgetHistogramaGraficoBarras->setPalette(pal);
49 }
50
51 void WidgetAnaliseExploracaoDados::habilitarEventos(bool habilitado) const
52 {
53     habilitado = habilitado && !_primeiraInicializacao;
54
55     ui.listWidgetAtributos->setEnabled(habilitado);
56     ui.treeWidgetTipos->setEnabled(habilitado);
57     ui.pushButtonRemover->setEnabled(habilitado);
58     ui.pushButtonTodos->setEnabled(habilitado);
59     ui.pushButtonInverter->setEnabled(habilitado);
60     ui.pushButtonEdicao->setEnabled(habilitado);
61     ui.pushButtonVisualizacao->setEnabled(habilitado);
62     ui.pushButtonSalvar->setEnabled(habilitado);
63 #ifndef TCC_RODOLFO
64     ui.pushButtonRemover->setEnabled(false); //Não se pode remover
           atributos dos dados do poço. Garantir consistência
65 #else
66     ui.pushButtonRemover->setEnabled(habilitado);
67 #endif
68 }
69
70 WidgetAnaliseExploracaoDados::~WidgetAnaliseExploracaoDados()
71 {
72     const void * address = static_cast<const void*>(&(*this));
73     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
74
75     if (_graficoBarras) delete _graficoBarras;
76     if (_histograma) delete _histograma;
77
78     ColetorEstatisticas::destruirInstancia();
79     TabelaDeRegistros::destruirInstancia();
80
81     for (vector<ThreadController*>::const_iterator citThread = _toDelete.
           cbegin();
```

```
82          citThread != _toDelete.cend();
83          citThread++)
84      {
85          delete *citThread;
86      }
87  }
88
89  void WidgetAnaliseExploracaoDados::inicialize()
90  {
91      if (_primeiraInicializacao)
92      {
93          connect(ui.listWidgetAtributos, SIGNAL(currentItemChanged(
                QListWidgetItem*,QListWidgetItem*)), this, SLOT(
                currentItemChangedListWidgetAtributos(QListWidgetItem*,
                QListWidgetItem*)));
94          connect(ui.treeWidgetTipos, SIGNAL(currentItemChanged(
                QTreeWidgetItem *,QTreeWidgetItem *)), this, SLOT(
                currentItemChangedTreeWidgetTipos(QTreeWidgetItem *,
                QTreeWidgetItem *)));
95          connect(ui.pushButtonRemover, SIGNAL(clicked()), this, SLOT(
                removerAtributoClick()));
96          connect(ui.pushButtonTodos, SIGNAL(clicked()), this, SLOT(
                selecionarTodosAtributosClick()));
97          connect(ui.pushButtonInverter, SIGNAL(clicked()), this, SLOT(
                inverterSelecaoClick()));
98          connect(ui.pushButtonEdicao, SIGNAL(clicked()), this, SLOT(
                edicaoClick()));
99          connect(ui.pushButtonVisualizacao, SIGNAL(clicked()), this, SLOT(
                visualizarGraficosClick()));
100 #ifndef TCC_RODOLFO
101         connect(ui.pushButtonSalvar, SIGNAL(clicked()), this, SLOT(
                salvarBDClick()));
102         ui.pushButtonRemover->setEnabled(false); //Não se pode remover
                atributos dos dados do poço. Garantir consistência
103 #else
104         connect(ui.pushButtonSalvar, SIGNAL(clicked()), this, SLOT(
                salvarArquivoClick()));
105 #endif
106     }
107
108     _primeiraInicializacao = false;
109
110     inicializeComponentes();
111 }
112
113 void WidgetAnaliseExploracaoDados::inicializeComponentes()
114 {
115     ui.labelTotalAtribsEdit->setText(QString::fromLatin1(Utilidades::
            dblToStdStr(TabelaDeRegistros::obterInstancia()->totalAtributos(),
            0).c_str()));
116     ui.labelTotalRegsEdit->setText(QString::fromLatin1(Utilidades::
            dblToStdStr(TabelaDeRegistros::obterInstancia()->totalRegistros(),
            0).c_str()));
117
118     const vector<string> &atributos = TabelaDeRegistros::obterInstancia()->
            atributos();
119
120     Populador::populeListWidget(ui.listWidgetAtributos, atributos);
121     Populador::populeArvoreTipos(ui.treeWidgetTipos, atributos,
            TabelaDeRegistros::obterInstancia()->tipos());
122
123     atualizeDadosComponentes(atributos.front());
124
125     habilitarEventos(true);
126     ui.pushButtonCarregar->setEnabled(true);
127 }
128
129 void WidgetAnaliseExploracaoDados::currentItemChangedListWidgetAtributos(
130     QListWidgetItem *current,
131     QListWidgetItem *previous)
132 {
133     if (current == previous)
134     {
135         return;
136     }
137
138     if (current)
139     {
140         atualizeDadosComponentes(current->text().toLatin1().data());
141     }
142 }
143
```

```
144 void WidgetAnaliseExploracaoDados::currentItemChangedTreeWidgetTipos(
145     QTreeWidgetItem *current,
146     QTreeWidgetItem *previous)
147 {
148     if (current == previous || current->parent() == NULL)
149     {
150         return;
151     }
152
153     if (current)
154     {
155         atualizeDadosComponentes(current->text(0).toLatin1().data());
156     }
157 }
158
159 void WidgetAnaliseExploracaoDados::atualizeDadosComponentes(const string &
        atributo)
160 {
161     ui.labelStatus->setText(QString::fromLatin1("Atualizando interface."));
162     ui.labelStatus->repaint();
163
164     populeListaEstatisticas(atributo);
165     atualizeDadosEstatisticaGeral(atributo);
166
167     ploteHistogramaGraficoBarras(atributo);
168
169     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
170     ui.labelStatus->repaint();
171 }
172
173 void WidgetAnaliseExploracaoDados::populeListaEstatisticas(const string &
        atributo) const
174 {
175     ui.tableWidgetEstatisticas->setRowCount(0);
176
177     vector<pair<string, double>> informacoes = vector<pair<string, double
            >>();
178     ColetorEstatisticas::obterInstancia()->estatisticasStr(atributo,
            informacoes);
179     Populador::populeTabelaEstatistica(ui.tableWidgetEstatisticas,
            informacoes);
180 }
181
182 void WidgetAnaliseExploracaoDados::atualizeDadosEstatisticaGeral(
183     const string &atributo) const
184 {
185     const ColetorEstatisticas::EstatisticaGeral &eg = ColetorEstatisticas::
            obterInstancia()->estatisticaGeralAtributo(atributo);
186
187     ui.labelNomeEdit->setText(QString::fromLatin1(atributo.c_str()));
188     ui.labelMissingEdit->setText(QString::fromLatin1(Utilidades::
            dblToStdStr(eg.missings, 0).c_str()));
189     ui.labelUnicosEdit->setText(QString::fromLatin1(Utilidades::dblToStdStr
            (eg.unicos, 0).c_str()));
190     ui.labelDistintosEdit->setText(QString::fromLatin1(Utilidades::
            dblToStdStr(eg.distintos, 0).c_str()));
191
192     switch (TabelaDeRegistros::obterInstancia()->tipo(atributo))
193     {
194     case QUANT_CONTINUO:
195         ui.labelTipoEdit->setText(QString::fromLatin1("Quantitativo
                Contínuo"));
196         break;
197     case QUANT_DISCRETO:
198         ui.labelTipoEdit->setText(QString::fromLatin1("Quantitativo
                Discreto"));
199         break;
200     case QUALITATIVO:
201         ui.labelTipoEdit->setText(QString::fromLatin1("Qualitativo"));
202         break;
203     default:
204         break;
205     }
206 }
207
208 void WidgetAnaliseExploracaoDados::ploteHistogramaGraficoBarras(const
        string &atributo)
209 {
210     switch (TabelaDeRegistros::obterInstancia()->tipo(atributo))
211     {
212     case QUANT_CONTINUO:
213         {
```

```
214                 ploteHistogramaQuantContinuo(atributo);
215             }
216             break;
217         case QUANT_DISCRETO:
218             {
219                 ploteGraficoBarrasQuantDiscreto(atributo);
220             }
221             break;
222         case QUALITATIVO:
223             {
224                 ploteGraficoBarrasQualitativo(atributo);
225             }
226             break;
227         }
228 }
229
230 void WidgetAnaliseExploracaoDados::ploteHistogramaQuantContinuo(const
        string &atributo)
231 {
232     vector<int> frequencias = vector<int>();
233     vector<ColetorEstatisticas::ClasseQuant> classes = vector<
            ColetorEstatisticas::ClasseQuant>();
234     GeradorGraficos::gerarHistogramaQuantContinuo(atributo, classes,
            frequencias);
235
236     if (classes.size() != frequencias.size())
237     {
238         return;
239     }
240
241     removerGraficoAtual();
242
243     _histograma = new HistogramaQuantContinuo(frequencias, classes
            .size(), ui.widgetHistogramaGraficoBarras); //destruido em
            removerHistogramaAtual e em ~WidgetAnaliseExploracaoDados
244     _histograma->setAxisScaleDraw(QwtPlot::xBottom, new EscalaNomes());
245
246     //QPalette p = _histograma->palette();
247     //p.setColor(QPalette::Window, Qt::red);
248     //_histograma->setPalette(p);
249     //_histograma->setCanvasBackground(QColor(Qt::green));
250
251     ui.widgetHistogramaGraficoBarras->layout()->addWidget(_histograma);
252
253     _histograma->setAxisTitle(QwtPlot::xBottom, QString::fromLatin1(
            atributo.c_str()));
254     _histograma->setAxisTitle(QwtPlot::yLeft, QString::fromLatin1("
            Frequência"));
255
256     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
257     ui.labelStatus->repaint();
258
259     ui.groupBoxHistogramaGraficoBarras->setTitle(QString::fromLatin1("
            Histograma"));
260 }
261
262 void WidgetAnaliseExploracaoDados::ploteGraficoBarrasQuantDiscreto(const
        string &atributo)
263 {
264     vector<int> frequencias = vector<int>();
265     vector<string> categorias = vector<string>();
266
267     GeradorGraficos::gerarGraficoBarrasQuantDiscreto(atributo, categorias,
            frequencias);
268
269     if (frequencias.size() != categorias.size())
270     {
271         return;
272     }
273
274     removerGraficoAtual();
275
276     ui.labelStatus->setText(QString::fromLatin1("Plotando o histograma...")
            );
277     ui.labelStatus->repaint();
278
279     _graficoBarras = new GraficoBarrasQualQuantDiscreto(categorias,
            frequencias, categorias.size(), ui.widgetHistogramaGraficoBarras);
280     ui.widgetHistogramaGraficoBarras->layout()->addWidget(_graficoBarras);
281
282     _graficoBarras->setAxisTitle(QwtPlot::xBottom, QString::fromLatin1(
            atributo.c_str()));
```

```
283         _graficoBarras->setAxisTitle(QwtPlot::yLeft, QString::fromLatin1("
                Frequência"));
284 
285         ui.labelStatus->setText(QString::fromLatin1("Pronto."));
286         ui.labelStatus->repaint();
287 
288         ui.groupBoxHistogramaGraficoBarras->setTitle(QString::fromLatin1("
                Gráfico de barras"));
289 }
290 void WidgetAnaliseExploracaoDados::ploteGraficoBarrasQualitativo(const
        string &atributo)
291 {
292         vector<int> frequencias = vector<int>();
293         vector<string> categorias = vector<string>();
294 
295         GeradorGraficos::gerarGraficoBarrasQualitativo(atributo, categorias,
                frequencias);
296 
297         if (frequencias.size() != categorias.size())
298         {
299             return;
300         }
301 
302         removerGraficoAtual();
303 
304         ui.labelStatus->setText(QString::fromLatin1("Plotando o gráfico..."));
305         ui.labelStatus->repaint();
306 
307         _graficoBarras = new GraficoBarrasQualQuantDiscreto(categorias,
                frequencias, categorias.size(), ui.widgetHistogramaGraficoBarras);
308         ui.widgetHistogramaGraficoBarras->layout()->addWidget(_graficoBarras);
309 
310         _graficoBarras->setAxisTitle(QwtPlot::xBottom, QString::fromLatin1(
                atributo.c_str()));
311         _graficoBarras->setAxisTitle(QwtPlot::yLeft, QString::fromLatin1("
                Frequência"));
312 
313         ui.labelStatus->setText(QString::fromLatin1("Pronto."));
314         ui.labelStatus->repaint();
315 
316         ui.groupBoxHistogramaGraficoBarras->setTitle(QString::fromLatin1("
                Gráfico de barras"));
317 }
318 
319 void WidgetAnaliseExploracaoDados::removerGraficoAtual()
320 {
321         if (_graficoBarras)
322         {
323             ui.widgetHistogramaGraficoBarras->layout()->removeWidget(
                    _graficoBarras);
324             delete _graficoBarras;
325             _graficoBarras = NULL;
326         }
327 
328         if (_histograma)
329         {
330             ui.widgetHistogramaGraficoBarras->layout()->removeWidget(
                    _histograma);
331             delete _histograma;
332             _histograma = NULL;
333         }
334 }
335 
336 void WidgetAnaliseExploracaoDados::removerAtributoClick()
337 {
338         if (TabelaDeRegistros::obterInstancia()->totalAtributos() == 1)
339         {
340             QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Não é possível remover o última atributo!"));
341             return;
342         }
343 
344         ui.labelStatus->setText(QString::fromLatin1("Removendo atributos
                selecionados..."));
345         ui.labelStatus->repaint();
346 
347         bool removerTodos = true;
348         bool removerNenhum = true;
349 
350         const int count = ui.listWidgetAtributos->count();
351 
352         for (int i = 0; i < count && (removerTodos || removerNenhum); ++i)
```

```
353     {
354         removerTodos = removerTodos && ui.listWidgetAtributos->item(i)->
                checkState() == Qt::Checked;
355         removerNenhum = removerNenhum && ui.listWidgetAtributos->item(i)->
                checkState() == Qt::Unchecked;
356     }
357
358     if (!(removerTodos || removerNenhum))
359     {
360         // Remove os atributos da tabela de registros
361
362         int totalRemovidos = 0;
363
364         for (int i = 0; i < count; ++i)
365         {
366             if (ui.listWidgetAtributos->item(i)->checkState() == Qt::
                    Checked)
367             {
368                 removerAtributo(ui.listWidgetAtributos->item(i)->text().
                        toStdString());
369                 ++totalRemovidos;
370             }
371         }
372
373         inicializeComponentes();
374     }
375     else
376     {
377         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Não é possível remover todos/nenhum os atributos!"
                ));
378     }
379
380     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
381     ui.labelStatus->repaint();
382 }
383
384 void WidgetAnaliseExploracaoDados::removerAtributo(const string &atributo)
        const
385 {
386     if (TabelaDeRegistros::obterInstancia()->totalAtributos() <= 1)
387     {
388         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Não é possível remover o último atributo!"));
389         return;
390     }
391
392     TabelaDeRegistros::obterInstancia()->removerAtributo(atributo);
393     ColetorEstatisticas::obterInstancia()->removerAtributo(atributo);
394 }
395
396 void WidgetAnaliseExploracaoDados::selecionarTodosAtributosClick() const
397 {
398     const int count = ui.listWidgetAtributos->count();
399
400     for (int i = 0; i < count; i++)
401     {
402         ui.listWidgetAtributos->item(i)->setCheckState(Qt::Checked);
403     }
404 }
405
406 void WidgetAnaliseExploracaoDados::inverterSelecaoClick() const
407 {
408     const int count = ui.listWidgetAtributos->count();
409
410     QListWidgetItem *item;
411
412     for (int i = 0; i < count; i++)
413     {
414         item = ui.listWidgetAtributos->item(i);
415
416         item->setCheckState(item->checkState() == Qt::Checked ? Qt::
                Unchecked : Qt::Checked);
417     }
418 }
419
420 void WidgetAnaliseExploracaoDados::edicaoClick()
421 {
422     ui.labelStatus->setText(QString::fromLatin1("Carregando..."));
423     ui.labelStatus->repaint();
424
425     vector<size_t> todosIndices = vector<size_t>();
```

```
426     TabelaDeRegistros::obterInstancia()->todosIndicesRegistros(todosIndices
            );
427
428     DialogEdicaoDados *dialogEdicaoDados = new DialogEdicaoDados(
            todosIndices, this); //deletado aqui abaixo;
429
430     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
431     ui.labelStatus->repaint();
432
433     dialogEdicaoDados->exec();
434
435     delete dialogEdicaoDados;
436
437     inicializeComponentes();
438 }
439
440 void WidgetAnaliseExploracaoDados::visualizarGraficosClick() const
441 {
442     ui.labelStatus->setText(QString::fromLatin1("Carregando..."));
443     ui.labelStatus->repaint();
444
445     DialogVisualizarGraficos *dialogVisualizarGraficos = new
            DialogVisualizarGraficos(); //deletado aqui abaixo
446
447     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
448     ui.labelStatus->repaint();
449
450     dialogVisualizarGraficos->exec();
451     delete dialogVisualizarGraficos;
452 }
453
454 void WidgetAnaliseExploracaoDados::salvarArquivoClick()
455 {
456     QString filtro = QString::fromLatin1("Arquivo Texto com Ponto-e-Vírgula
            de Delimitador (*.csv)");
457     const QString diretorioArquivo = QFileDialog::getSaveFileName(this,"
            Salvar Arquivo", QDir::currentPath(), QString::fromLatin1(
458         "Arquivo Texto com Ponto-e-Vírgula de Delimitador (*.csv);;Arquivo
                Texto com Tabulação de Delimitador (*.csv)"),
459         &filtro);
460
461     string delimitador;
462
463     if (diretorioArquivo.isEmpty())
464     {
465         return;
466     }
467
468     if (filtro == QString::fromLatin1("Arquivo Texto com Ponto-e-Vírgula de
            Delimitador (*.csv)"))
469     {
470         delimitador = ";";
471     }
472     else if (filtro == QString::fromLatin1("Arquivo Texto com Tabulação de
            Delimitador (*.csv)"))
473     {
474         delimitador = "\t";
475     }
476     else
477     {
478         return;
479     }
480
481     if (!salveTabelaParaArqTexto((string) diretorioArquivo.toLatin1().data
            (), delimitador))
482     {
483         QMessageBox::critical(NULL, QString::fromLatin1("Erro"), QString::
                fromLatin1("Erro ao salvar arquivo!"));
484     }
485     else
486     {
487         QMessageBox::information(NULL, QString::fromLatin1("Sucesso"),
                QString::fromLatin1("Arquivo salvo com sucesso!"));
488     }
489 }
490
491 bool WidgetAnaliseExploracaoDados::salveTabelaParaArqTexto(
492     const string &arquivo,
493     const string &delimitador) const
494 {
495     ui.labelStatus->setText(QString::fromLatin1("Salvando em arquivo..."));
496     ui.labelStatus->repaint();
```

```
497
498     try
499     {
500         ofstream arqSaida(arquivo.c_str());
501
502         const TabelaDeRegistros::MapRegistros &registros =
                TabelaDeRegistros::obterInstancia()->registros();
503         const vector<string> &atributos = TabelaDeRegistros::obterInstancia
                ()->atributos();
504
505         const size_t countColunas = TabelaDeRegistros::obterInstancia()->
                totalAtributos();
506
507         for (size_t j = 0; j < countColunas; j++)
508         {
509             arqSaida << atributos[j];
510
511             if (j < (countColunas - 1))
512             {
513                 arqSaida << delimitador;
514             }
515         }
516
517         arqSaida << endl;
518
519         for (TabelaDeRegistros::MapRegistros::const_iterator citRegistro =
                registros.cbegin();
520             citRegistro != registros.cend();
521             citRegistro++)
522         {
523             const Registro &registro= citRegistro->second;
524
525             for (size_t j = 0; j < countColunas; ++j)
526             {
527                 arqSaida << registro.retorneValor(j)->valorStr();
528
529                 if (j < (countColunas - 1))
530                 {
531                     arqSaida << delimitador;
532                 }
533             }
534             arqSaida << endl;
535         }
536     }
537     catch(...)
538     {
539     }
540
541     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
542     ui.labelStatus->repaint();
543
544     return true;
545 }
546
547 #ifndef TCC_RODOLFO
548 void WidgetAnaliseExploracaoDados::carregarBDClick()
549 {
550     vector<shared_ptr<PocoLog>> vetor = IntegracaoTreinamento::obterPocos()
            ;
551     EscolhaPoco *carregar = new EscolhaPoco(vetor); //é deletado no accept
            da janela carregar
552     connect(carregar, SIGNAL(itemSelected(shared_ptr<PocoLog>)), this, SLOT
            (pocoSelecionado(shared_ptr<PocoLog>)));
553
554     carregar->exec();
555 }
556
557 void WidgetAnaliseExploracaoDados::pocoSelecionado(shared_ptr<PocoLog> p)
558 {
559     habilitarEventos(false);
560     ui.pushButtonCarregar->setEnabled(false);
561
562     ui.labelStatus->setText(QString::fromLatin1("Carregando do BD..."));
563     ui.labelStatus->repaint();
564
565     bool sucesso = false;
566
567     vector<vector<ValorBase *const>> colunas = vector<vector<ValorBase *
            const>>();
568     vector<TipoDado_AT> tipos = vector<TipoDado_AT>();
569     vector<string> atributos = vector<string>();
570
```

```
571     try
572     {
573         comuns_eng::Dados dadosBanco = IntegracaoTreinamento::
                obterRegistrosParaAnaliseDeDados(p);
574
575         vector<string> atributosQuant = dadosBanco.nomesColunasDadosQuant()
                ;
576         vector<string> atributosQuali = dadosBanco.nomesColunasDadosQuali()
                ;
577
578         boost::numeric::ublas::matrix<double> dadosQuant = dadosBanco.
                dadosQuant();
579         boost::numeric::ublas::matrix<string> dadosQuali = dadosBanco.
                dadosQuali();
580
581         const size_t totalLinhasQuant = dadosQuant.size1();
582         const size_t totalAtributosQuant = dadosQuant.size2();
583         const size_t totalAtributosQuant2 = atributosQuant.size();
584
585         const size_t totalLinhasQuali = dadosQuali.size1();
586         const size_t totalAtributosQuali = dadosQuali.size2();
587         const size_t totalAtributosQuali2 = atributosQuali.size();
588
589         if (totalAtributosQuant != 0 && totalAtributosQuali != 0 &&
                totalLinhasQuant != totalLinhasQuali)
590         {
591             QMessageBox::critical(this, QString::fromLatin1("Erro"),
                    QString::fromLatin1("Inconsistência nos dados!"));
592         }
593         else if (totalLinhasQuant == 0 && totalLinhasQuali == 0)
594         {
595             QMessageBox::critical(this, QString::fromLatin1("Erro"),
                    QString::fromLatin1("Não há nenhum registro!"));
596         }
597         else
598         {
599             atributos.reserve(totalAtributosQuant + totalAtributosQuali);
600             colunas.reserve(totalAtributosQuant + totalAtributosQuali);
601
602             for(size_t colunaIdx = 0; colunaIdx < totalAtributosQuant;
                    colunaIdx++)
603             {
604                 atributos.push_back(atributosQuant[colunaIdx]);
605                 tipos.push_back(QUANT_CONTINUO);
606
607                 colunas.push_back(vector<ValorBase *const>());
608
609                 vector<ValorBase *const> &valoresColuna = colunas.back();
610                 valoresColuna.reserve(totalLinhasQuant);
611
612                 for(size_t linhaIdx = 0; linhaIdx < totalLinhasQuant;
                        linhaIdx++)
613                 {
614                     valoresColuna.push_back(TabelaDeRegistros::
                            gerarValorQuant(dadosQuant(linhaIdx, colunaIdx)));
615                 }
616             }
617
618             for(size_t colunaIdx = 0; colunaIdx < totalAtributosQuali;
                    colunaIdx++)
619             {
620                 atributos.push_back(atributosQuali[colunaIdx]);
621                 tipos.push_back(QUALITATIVO);
622
623                 colunas.push_back(vector<ValorBase *const>());
624
625                 vector<ValorBase *const> &valoresColuna = colunas.back();
626                 valoresColuna.reserve(totalLinhasQuali);
627
628                 for(size_t linhaIdx = 0; linhaIdx < totalLinhasQuali;
                        linhaIdx++)
629                 {
630                     valoresColuna.push_back(TabelaDeRegistros::gerarValor(
                            QUALITATIVO, dadosQuali(linhaIdx, colunaIdx)));
631                 }
632             }
633
634             sucesso = true;
635         }
636     }
637     catch (...)
638     {
```

```
639             QMessageBox::critical(this, QString::fromLatin1("Erro"), QString::
                    fromLatin1("Ocorreu um erro ao tentar carregar os dados!"));
640         }
641
642         ui.labelStatus->setText(QString::fromLatin1("Pronto."));
643         ui.labelStatus->repaint();
644
645         QMessageBox msgQuestion(QMessageBox::Question, "Opcional", QString::
                fromLatin1("Deseja abrir o assistente de carregamento dos dados?")
                , QMessageBox::Yes|QMessageBox::No);
646         msgQuestion.button(QMessageBox::Yes)->setText(QString::fromLatin1("Sim"
                ));
647         msgQuestion.button(QMessageBox::No)->setText(QString::fromLatin1("Não")
                );
648
649         if (sucesso)
650         {
651             if (msgQuestion.exec() == QMessageBox::Yes)
652             {
653                 DialogImportacaoDados *dialogImportacaoDados = new
                        DialogImportacaoDados(this);
654                 //dialogImportacaoDados->setAttribute(Qt::WA_DeleteOnClose);
                        nao descomentar, problemas em algumas máquinas
655
656                 dialogImportacaoDados->setDados(atributos, tipos, colunas);
657
658                 dialogImportacaoDados->exec();
659                 //delete dialogImportacaoDados; nao descomentar, problemas em
                        algumas máquinas
660
661                 ui.labelStatus->setText(QString::fromLatin1("Pronto."));
662                 ui.labelStatus->repaint();
663
664                 if (TabelaDeRegistros::obterInstancia()->totalRegistros() > 0)
665                 {
666                     inicialize();
667                 }
668
669                 ui.progressBar->setValue(0);
670             }
671             else
672             {
673                 ThreadWorkerColetarEstatisticas* workerColetarEstatisticas =
                        new ThreadWorkerColetarEstatisticas(atributos, tipos,
                        colunas, this); //deletado quando destroi
                        _threadController ou quando _threadController seta um novo
                         worker
674                 ThreadController *tc = new ThreadController();
675                 connect(tc, SIGNAL(execucaoFinalizada(ThreadController*)), this
                        , SLOT(deleteController(ThreadController*)));
676                 tc->setarWorker(workerColetarEstatisticas);
677                 tc->iniciarExecucaoWorker(this);
678             }
679         }
680         else
681         {
682             habilitarEventos(true);
683             ui.pushButtonCarregar->setEnabled(true);
684         }
685 }
686
687
688 void WidgetAnaliseExploracaoDados::salvarBDClick()
689 {
690     bool temMissingOuInvalido = false;
691
692     const TabelaDeRegistros::MapRegistros &regs = TabelaDeRegistros::
            obterInstancia()->registros();
693
694     for (TabelaDeRegistros::MapRegistros::const_iterator citRegistro = regs
            .cbegin();
695         citRegistro != regs.cend() && !temMissingOuInvalido;
696         citRegistro++)
697     {
698         const Registro &registro= citRegistro->second;
699         if (registro.algumValorInvalidoOuMissing())
700         {
701             temMissingOuInvalido = true;
702         }
703     }
704
705     if (temMissingOuInvalido)
```

```
706     {
707         QMessageBox msgQuestion(QMessageBox::Question, "Aviso", QString::
                fromLatin1(
708             "Existem valores inválidos ou missings nos dados!\n"
709             "Os registros que contenham valores missings ou inválidos não
                    serão salvos."
710             "Deseja continuar mesmo assim?\n"), QMessageBox::Yes|
                    QMessageBox::No);
711         msgQuestion.button(QMessageBox::Yes)->setText(QString::fromLatin1("
                Sim"));
712         msgQuestion.button(QMessageBox::No)->setText(QString::fromLatin1("
                Não"));
713 
714         if (msgQuestion.exec() == QMessageBox::No)
715         {
716             return;
717         }
718     }
719 
720     SalvarNoBD *salvar = new SalvarNoBD("", "", this);
721     connect(salvar, SIGNAL(itemSavedAs(string, string)), this, SLOT(
            salvarBD(string, string)));
722     salvar->exec();
723     //delete salvar;
724 }
725 
726 void WidgetAnaliseExploracaoDados::salvarBD(string codigoPetro, string
        descricao)
727 {
728     if (DaoPoco::existe(codigoPetro))
729     {
730         string msg = "Poço " + codigoPetro + " já existe! Você deve salvar
                com outro nome.";
731 
732         QMessageBox::critical(this, QString::fromLatin1("Erro"), QString::
                fromLatin1(msg.c_str()));
733 
734         SalvarNoBD *salvar = new SalvarNoBD(codigoPetro, "", this);
735         connect(salvar, SIGNAL(itemSavedAs(string, string)), this, SLOT(
                salvarBD(string, string)));
736         salvar->exec();
737         //delete salvar;
738         return;
739     }
740 
741     habilitarEventos(false);
742     ui.pushButtonCarregar->setEnabled(false);
743 
744     ui.labelStatus->setText(QString::fromLatin1("Salvando no Banco de dados
            ..."));
745     ui.labelStatus->repaint();
746 
747     try
748     {
749         std::vector<std::vector<std::string>> registros;
750 
751         ThreadWorkerSalvarBD* workerSalvar = new ThreadWorkerSalvarBD(
                TabelaDeRegistros::obterInstancia(), codigoPetro, this); //
                deletado quando destroi _threadController ou quando
                _threadController seta um novo worker
752         ThreadController *tc = new ThreadController();
753         connect(tc, SIGNAL(execucaoFinalizada(ThreadController*)), this,
                SLOT(deleteController(ThreadController*)));
754         tc->setarWorker(workerSalvar);
755         tc->iniciarExecucaoWorker(this);
756 
757     }
758     catch (...)
759     {
760         QMessageBox::critical(this, QString::fromLatin1("Erro"), QString::
                fromLatin1("Erro ao salvar no Banco de dados!"));
761 
762         habilitarEventos(true);
763         ui.pushButtonCarregar->setEnabled(true);
764     }
765 }
766 
767 void WidgetAnaliseExploracaoDados::fimSalvarBD(bool sucesso)
768 {
769     if (!sucesso)
770     {
771         QMessageBox::critical(this, QString::fromLatin1("Erro"), QString::
```

```
                fromLatin1("Erro ao salvar no Banco de dados!"));
772     }
773     else
774     {
775         QMessageBox::information(this, QString::fromLatin1("Sucesso"),
                QString::fromLatin1("Poço salvo com sucesso!"));
776     }
777
778     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
779     ui.labelStatus->repaint();
780
781     habilitarEventos(true);
782     ui.pushButtonCarregar->setEnabled(true);
783
784     ui.progressBar->setValue(0);
785 }
786 #endif
787
788 void WidgetAnaliseExploracaoDados::carregarArquivoClick()
789 {
790     DialogImportacaoDados *dialogImportacaoDados = new
            DialogImportacaoDados(this);
791     //dialogImportacaoDados->setAttribute(Qt::WA_DeleteOnClose); nao
            descomentar, problemas em algumas máquinas
792
793     dialogImportacaoDados->exec();
794     //delete dialogImportacaoDados; nao descomentar, problemas em algumas
            máquinas
795
796     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
797     ui.labelStatus->repaint();
798
799     if (TabelaDeRegistros::obterInstancia()->totalRegistros() > 0)
800     {
801         inicialize();
802     }
803
804     ui.progressBar->setValue(0);
805 }
806
807 void WidgetAnaliseExploracaoDados::fimColetaEstatisticas(const vector<
        string> *atributos)
808 {
809     delete atributos;
810
811     ui.labelStatus->setText(QString::fromLatin1("Pronto."));
812     ui.labelStatus->repaint();
813
814     inicialize();
815
816     ui.progressBar->setValue(0);
817 }
818
819 void WidgetAnaliseExploracaoDados::onUpdateProgresso(int *porcentagem)
820 {
821     ui.progressBar->setValue(*porcentagem);
822     delete porcentagem;
823 }
824
825 void WidgetAnaliseExploracaoDados::deleteController(ThreadController *tc)
826 {
827     _toDelete.push_back(tc);
828 }
```

Listagem B.72: widget_analise_exploracao_dados.cpp

```
1  #ifndef WIDGET_ANALISE_EXPLORACAO_DADOS_H
2  #define WIDGET_ANALISE_EXPLORACAO_DADOS_H
3
4  #include "ui_widget_analise_exploracao_dados.h"
5
6  #include "widget_progresso.h"
7
8  #include "thread_controller.h"
9  #include "histograma_quant_continuo.h"
10 #include "grafico_barras_qual_quant_discreto.h"
11
12 #include "enums_at.h"
13
```

```
14 #ifndef TCC_RODOLFO
15 #include "poco_log.h"
16 #endif
17
18 using namespace std;
19
20 class WidgetAnaliseExploracaoDados : public WidgetProgresso
21 {
22     Q_OBJECT
23
24 public:
25     WidgetAnaliseExploracaoDados(QWidget *parent = NULL);
26     ~WidgetAnaliseExploracaoDados();
27
28 private:
29     Ui::WidgetAnaliseExploracaoDados ui;
30     HistogramaQuantContinuo *_histograma;
31     GraficoBarrasQualQuantDiscreto *_graficoBarras;
32     bool _primeiraInicializacao;
33     vector<ThreadController*> _toDelete;
34
35     void habilitarEventos(const bool habilitado) const;
36     void atualizeDadosComponentes(const string &atributo);
37     void populeListaEstatisticas(const string &atributo) const;
38     void atualizeDadosEstatisticaGeral(const string &atributo) const;
39     void ploteHistogramaGraficoBarras(const string &atributo);
40     void ploteHistogramaQuantContinuo(const string &atributo);
41     void ploteGraficoBarrasQuantDiscreto(const string &atributo);
42     void ploteGraficoBarrasQualitativo(const string &atributo);
43     void removerGraficoAtual();
44     void removerAtributo(const string &atributo) const;
45     bool salveTabelaParaArqTexto(
46         const string &arquivo,
47         const string &delimitador) const;
48     bool desejaSalvar(const char *s) const;
49     void inicialize();
50     void inicializeComponentes();
51     void coletaEstatisticasTipo() const;
52
53 private slots:
54     void currentItemChangedListWidgetAtributos(QListWidgetItem *current,
           QListWidgetItem *previous);
55     void currentItemChangedTreeWidgetTipos(QTreeWidgetItem *current,
           QTreeWidgetItem *previous);
56     void removerAtributoClick();
57     void selecionarTodosAtributosClick() const;
58     void inverterSelecaoClick() const;
59     void edicaoClick();
60     void visualizarGraficosClick() const;
61 #ifndef TCC_RODOLFO
62     void carregarBDClick();
63     void salvarBDClick();
64 #endif
65     void carregarArquivoClick();
66     void salvarArquivoClick();
67
68 public slots:
69 #ifndef TCC_RODOLFO
70     virtual void onUpdateProgresso(StatusThread*) {};
71     void pocoSelecionado(shared_ptr<PocoLog> p);
72     void salvarBD(string codigoPetro, string descricao);
73     void fimSalvarBD(bool sucesso);
74 #endif
75     virtual void onUpdateProgresso(int *porcentagem);
76     void deleteController(ThreadController *tc);
77     void fimColetaEstatisticas(const vector<string> *atributos);
78 };
79
80 #endif // WIDGET_ANALISE_EXPLORACAO_DADOS_H
```

Listagem B.73: widget_analise_exploracao_dados.h

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <ui version="4.0">
3  <class>WidgetAnaliseExploracaoDados</class>
4  <widget class="QWidget" name="WidgetAnaliseExploracaoDados">
5   <property name="geometry">
6    <rect>
7     <x>0</x>
```

```
 8       <y>0</y>
 9       <width>951</width>
10       <height>744</height>
11      </rect>
12     </property>
13     <property name="minimumSize">
14      <size>
15       <width>825</width>
16       <height>635</height>
17      </size>
18     </property>
19     <property name="windowTitle">
20      <string>Análise e Tratamento dos Dados</string>
21     </property>
22     <layout class="QVBoxLayout" name="verticalLayout_4">
23      <item>
24       <widget class="QLabel" name="label_2">
25        <property name="sizePolicy">
26         <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
27          <horstretch>0</horstretch>
28          <verstretch>0</verstretch>
29         </sizepolicy>
30        </property>
31        <property name="font">
32         <font>
33          <pointsize>17</pointsize>
34         </font>
35        </property>
36        <property name="text">
37         <string>Análise e Tratamento dos Dados</string>
38        </property>
39       </widget>
40      </item>
41      <item>
42       <layout class="QHBoxLayout" name="horizontalLayout_2">
43        <item>
44         <widget class="QSplitter" name="splitter">
45          <property name="sizePolicy">
46           <sizepolicy hsizetype="Fixed" vsizetype="Expanding">
47            <horstretch>0</horstretch>
48            <verstretch>0</verstretch>
49           </sizepolicy>
50          </property>
51          <property name="orientation">
52           <enum>Qt::Vertical</enum>
53          </property>
54          <widget class="QGroupBox" name="groupBoxInfoGerais">
55           <property name="title">
56            <string>Informações gerais</string>
57           </property>
58           <layout class="QHBoxLayout" name="horizontalLayout_4">
59            <item>
60             <layout class="QVBoxLayout" name="verticalLayout_2">
61              <property name="sizeConstraint">
62               <enum>QLayout::SetFixedSize</enum>
63              </property>
64              <item>
65               <widget class="QProgressBar" name="progressBar">
66                <property name="sizePolicy">
67                 <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
68                  <horstretch>0</horstretch>
69                  <verstretch>0</verstretch>
70                 </sizepolicy>
71                </property>
72                <property name="minimumSize">
73                 <size>
74                  <width>0</width>
75                  <height>0</height>
76                 </size>
77                </property>
78                <property name="maximumSize">
79                 <size>
80                  <width>16777215</width>
81                  <height>16777215</height>
82                 </size>
83                </property>
84                <property name="value">
85                 <number>0</number>
86                </property>
87               </widget>
88              </item>
89              <item>
```

```
90              <widget class="QLabel" name="labelStatus">
91               <property name="sizePolicy">
92                <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
93                 <horstretch>0</horstretch>
94                 <verstretch>0</verstretch>
95                </sizepolicy>
96               </property>
97               <property name="minimumSize">
98                <size>
99                 <width>0</width>
100                <height>0</height>
101               </size>
102              </property>
103              <property name="maximumSize">
104               <size>
105                <width>16777215</width>
106                <height>16777215</height>
107               </size>
108              </property>
109              <property name="text">
110               <string>Pronto.</string>
111              </property>
112             </widget>
113            </item>
114           </layout>
115          </item>
116          <item>
117           <spacer name="horizontalSpacer_3">
118            <property name="orientation">
119             <enum>Qt::Horizontal</enum>
120            </property>
121            <property name="sizeType">
122             <enum>QSizePolicy::Fixed</enum>
123            </property>
124            <property name="sizeHint" stdset="0">
125             <size>
126              <width>20</width>
127              <height>20</height>
128             </size>
129            </property>
130           </spacer>
131          </item>
132          <item>
133           <layout class="QFormLayout" name="formLayout_4">
134            <property name="sizeConstraint">
135             <enum>QLayout::SetNoConstraint</enum>
136            </property>
137            <property name="horizontalSpacing">
138             <number>10</number>
139            </property>
140            <property name="verticalSpacing">
141             <number>15</number>
142            </property>
143            <item row="0" column="0">
144             <widget class="QLabel" name="labelTotalRegs">
145              <property name="text">
146               <string>Total de registros:</string>
147              </property>
148             </widget>
149            </item>
150            <item row="0" column="1">
151             <widget class="QLabel" name="labelTotalRegsEdit">
152              <property name="text">
153               <string/>
154              </property>
155             </widget>
156            </item>
157            <item row="1" column="0">
158             <widget class="QLabel" name="labelTotalAtribs">
159              <property name="text">
160               <string>Total de atributos:</string>
161              </property>
162             </widget>
163            </item>
164            <item row="1" column="1">
165             <widget class="QLabel" name="labelTotalAtribsEdit">
166              <property name="text">
167               <string/>
168              </property>
169             </widget>
170            </item>
171           </layout>
```

```
172           </item>
173          </layout>
174         </widget>
175         <widget class="QGroupBox" name="groupBoxAtributos">
176          <property name="minimumSize">
177           <size>
178            <width>0</width>
179            <height>0</height>
180           </size>
181          </property>
182          <property name="maximumSize">
183           <size>
184            <width>16777215</width>
185            <height>16777215</height>
186           </size>
187          </property>
188          <property name="title">
189           <string>Atributos</string>
190          </property>
191          <layout class="QVBoxLayout" name="verticalLayout">
192           <item>
193            <layout class="QHBoxLayout" name="horizontalLayout">
194             <item>
195              <widget class="QPushButton" name="pushButtonTodos">
196               <property name="sizePolicy">
197                <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
198                 <horstretch>0</horstretch>
199                 <verstretch>0</verstretch>
200                </sizepolicy>
201               </property>
202               <property name="text">
203                <string>Todos</string>
204               </property>
205              </widget>
206             </item>
207             <item>
208              <widget class="QPushButton" name="pushButtonRemover">
209               <property name="sizePolicy">
210                <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
211                 <horstretch>0</horstretch>
212                 <verstretch>0</verstretch>
213                </sizepolicy>
214               </property>
215               <property name="text">
216                <string>Remover</string>
217               </property>
218              </widget>
219             </item>
220             <item>
221              <widget class="QPushButton" name="pushButtonInverter">
222               <property name="sizePolicy">
223                <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
224                 <horstretch>0</horstretch>
225                 <verstretch>0</verstretch>
226                </sizepolicy>
227               </property>
228               <property name="text">
229                <string>Inverter</string>
230               </property>
231              </widget>
232             </item>
233            </layout>
234           </item>
235           <item>
236            <widget class="QListWidget" name="listWidgetAtributos">
237             <property name="sizePolicy">
238              <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
239               <horstretch>0</horstretch>
240               <verstretch>0</verstretch>
241              </sizepolicy>
242             </property>
243            </widget>
244           </item>
245          </layout>
246         </widget>
247         <widget class="QGroupBox" name="groupBoxTipos">
248          <property name="title">
249           <string>Tipos</string>
250          </property>
251          <layout class="QHBoxLayout" name="horizontalLayout_3">
252           <item>
253            <widget class="QTreeWidget" name="treeWidgetTipos">
```

```
254             <property name="sizePolicy">
255              <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
256               <horstretch>0</horstretch>
257               <verstretch>0</verstretch>
258              </sizepolicy>
259             </property>
260             <column>
261              <property name="text">
262               <string/>
263              </property>
264             </column>
265            </widget>
266           </item>
267          </layout>
268         </widget>
269         <widget class="QGroupBox" name="groupBoxOpcoes">
270          <property name="minimumSize">
271           <size>
272            <width>0</width>
273            <height>0</height>
274           </size>
275          </property>
276          <property name="title">
277           <string>Opções</string>
278          </property>
279          <layout class="QGridLayout" name="gridLayout">
280           <item row="0" column="0">
281            <widget class="QPushButton" name="pushButtonCarregar">
282             <property name="sizePolicy">
283              <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
284               <horstretch>0</horstretch>
285               <verstretch>0</verstretch>
286              </sizepolicy>
287             </property>
288             <property name="text">
289              <string>Carregar</string>
290             </property>
291            </widget>
292           </item>
293           <item row="0" column="1">
294            <widget class="QPushButton" name="pushButtonSalvar">
295             <property name="sizePolicy">
296              <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
297               <horstretch>0</horstretch>
298               <verstretch>0</verstretch>
299              </sizepolicy>
300             </property>
301             <property name="text">
302              <string>Salvar</string>
303             </property>
304            </widget>
305           </item>
306           <item row="1" column="0">
307            <widget class="QPushButton" name="pushButtonEdicao">
308             <property name="sizePolicy">
309              <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
310               <horstretch>0</horstretch>
311               <verstretch>0</verstretch>
312              </sizepolicy>
313             </property>
314             <property name="text">
315              <string>Edição</string>
316             </property>
317            </widget>
318           </item>
319           <item row="1" column="1">
320            <widget class="QPushButton" name="pushButtonVisualizacao">
321             <property name="sizePolicy">
322              <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
323               <horstretch>0</horstretch>
324               <verstretch>0</verstretch>
325              </sizepolicy>
326             </property>
327             <property name="text">
328              <string>Visualização</string>
329             </property>
330            </widget>
331           </item>
332          </layout>
333         </widget>
334        </widget>
335       </item>
```

```
336       <item>
337        <widget class="QSplitter" name="splitter_2">
338         <property name="enabled">
339          <bool>true</bool>
340         </property>
341         <property name="orientation">
342          <enum>Qt::Vertical</enum>
343         </property>
344         <property name="childrenCollapsible">
345          <bool>false</bool>
346         </property>
347         <widget class="QGroupBox" name="groupBoxAtributoSelecionado">
348          <property name="title">
349           <string>Atributo Selecionado</string>
350          </property>
351          <layout class="QVBoxLayout" name="verticalLayout_5">
352           <item>
353            <layout class="QHBoxLayout" name="horizontalLayout_6">
354             <item>
355              <layout class="QFormLayout" name="formLayout">
356               <property name="fieldGrowthPolicy">
357                <enum>QFormLayout::AllNonFixedFieldsGrow</enum>
358               </property>
359               <item row="0" column="0">
360                <widget class="QLabel" name="labelNome">
361                 <property name="text">
362                  <string>Nome:</string>
363                 </property>
364                </widget>
365               </item>
366               <item row="0" column="1">
367                <widget class="QLabel" name="labelNomeEdit">
368                 <property name="text">
369                  <string/>
370                 </property>
371                </widget>
372               </item>
373               <item row="1" column="0">
374                <widget class="QLabel" name="labelDistintos">
375                 <property name="text">
376                  <string>Total de valores distintos:</string>
377                 </property>
378                </widget>
379               </item>
380               <item row="1" column="1">
381                <widget class="QLabel" name="labelDistintosEdit">
382                 <property name="text">
383                  <string/>
384                 </property>
385                </widget>
386               </item>
387               <item row="2" column="0">
388                <widget class="QLabel" name="labelUnicos">
389                 <property name="text">
390                  <string>Total de valores únicos:</string>
391                 </property>
392                </widget>
393               </item>
394               <item row="2" column="1">
395                <widget class="QLabel" name="labelUnicosEdit">
396                 <property name="text">
397                  <string/>
398                 </property>
399                </widget>
400               </item>
401              </layout>
402             </item>
403             <item>
404              <spacer name="horizontalSpacer_2">
405               <property name="orientation">
406                <enum>Qt::Horizontal</enum>
407               </property>
408               <property name="sizeType">
409                <enum>QSizePolicy::Fixed</enum>
410               </property>
411               <property name="sizeHint" stdset="0">
412                <size>
413                 <width>40</width>
414                 <height>20</height>
415                </size>
416               </property>
417              </spacer>
```

```
418             </item>
419             <item>
420              <layout class="QFormLayout" name="formLayout_2">
421               <item row="0" column="0">
422                <widget class="QLabel" name="labelTipo">
423                 <property name="text">
424                  <string>Tipo:</string>
425                 </property>
426                </widget>
427               </item>
428               <item row="0" column="1">
429                <widget class="QLabel" name="labelTipoEdit">
430                 <property name="text">
431                  <string/>
432                 </property>
433                </widget>
434               </item>
435               <item row="1" column="0">
436                <widget class="QLabel" name="labelMissing">
437                 <property name="text">
438                  <string>Total de valores missings:</string>
439                 </property>
440                </widget>
441               </item>
442               <item row="1" column="1">
443                <widget class="QLabel" name="labelMissingEdit">
444                 <property name="text">
445                  <string/>
446                 </property>
447                </widget>
448               </item>
449              </layout>
450             </item>
451             <item>
452              <spacer name="horizontalSpacer">
453               <property name="orientation">
454                <enum>Qt::Horizontal</enum>
455               </property>
456               <property name="sizeHint" stdset="0">
457                <size>
458                 <width>40</width>
459                 <height>20</height>
460                </size>
461               </property>
462              </spacer>
463             </item>
464            </layout>
465           </item>
466           <item>
467            <widget class="QTableWidget" name="tableWidgetEstatisticas">
468             <column>
469              <property name="text">
470               <string/>
471              </property>
472             </column>
473             <column>
474              <property name="text">
475               <string/>
476              </property>
477             </column>
478            </widget>
479           </item>
480          </layout>
481         </widget>
482         <widget class="QGroupBox" name="groupBoxHistogramaGraficoBarras">
483          <property name="title">
484           <string>Histograma</string>
485          </property>
486          <layout class="QVBoxLayout" name="verticalLayout_3">
487           <item>
488            <widget class="QWidget" name="widgetHistogramaGraficoBarras"
                    native="true">
489             <layout class="QVBoxLayout" name="verticalLayout_6">
490              <property name="leftMargin">
491               <number>0</number>
492              </property>
493              <property name="topMargin">
494               <number>0</number>
495              </property>
496              <property name="rightMargin">
497               <number>0</number>
498              </property>
```

```
499              <property name="bottomMargin">
500               <number>0</number>
501              </property>
502             </layout>
503            </widget>
504           </item>
505          </layout>
506         </widget>
507        </widget>
508       </item>
509      </layout>
510     </item>
511    </layout>
512   </widget>
513   <resources/>
514   <connections/>
515 </ui>
```

Listagem B.74: widget_analise_exploracao_dados.ui

```
1  #include "widget_categorias_qual.h"
2
3  #include "populador.h"
4  #include "coletor_estatisticas.h"
5
6  #include <QMessageBox>
7
8  #include <QDebug>
9
10 WidgetCategoriasQual::WidgetCategoriasQual(
11     const vector<string> &atributos,
12     QWidget *parent)
13     : QWidget(parent)
14 {
15     inicialize(atributos);
16 }
17
18 WidgetCategoriasQual::WidgetCategoriasQual(
19     const string &atributo,
20     QWidget *parent)
21     : QWidget(parent)
22 {
23     vector<string> atributos = vector<string>();
24     atributos.push_back(atributo);
25
26     inicialize(atributos);
27 }
28
29 WidgetCategoriasQual::~WidgetCategoriasQual()
30 {
31     const void * address = static_cast<const void*>(&(*this));
32     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
33 }
34
35 void WidgetCategoriasQual::inicialize(const vector<string> &atributos)
36 {
37     ui.setupUi(this);
38
39     _atributos = atributos;
40
41     Populador::populeComboBox(ui.comboBoxAtributos, _atributos);
42
43     connect(ui.comboBoxAtributos, SIGNAL(currentIndexChanged(const QString
           &)), this, SLOT(currentIndexChangedComboBoxAtributos(const QString
           &)));
44     if (!atributos.empty())
45     {
46         currentIndexChangedComboBoxAtributos(ui.comboBoxAtributos->
               currentText());
47     }
48
49     connect(ui.listWidgetValores, SIGNAL(itemChanged(QListWidgetItem*)),
           this, SLOT(itemChanged(QListWidgetItem*)));
50 }
51
52 void WidgetCategoriasQual::setHabilitadoComboBoxAtributos(const bool
       habilitado)
53 {
54     ui.comboBoxAtributos->setEnabled(habilitado);
```

```
55 }
56
57 void WidgetCategoriasQual::setAtributoAtualComboBoxAtributos(const string &
       atributo)
58 {
59     vector<string>::const_iterator citAtributo = std::find(_atributos.
           cbegin(), _atributos.cend(), atributo);
60
61     if (citAtributo != _atributos.cend())
62     {
63         ui.comboBoxAtributos->setCurrentIndex(citAtributo - _atributos.
               cbegin());
64     }
65 }
66
67 void WidgetCategoriasQual::currentIndexChangedComboBoxAtributos(const
       QString &text)
68 {
69     const string atributoSelecionado = text.toLatin1().data();
70     vector<string> categoriasAtributoQualitativo = vector<string>();
71     ColetorEstatisticas::obterInstancia()->todasCategoriasAtributoQual(
           atributoSelecionado, categoriasAtributoQualitativo);
72
73     Populador::populeListWidget(ui.listWidgetValores,
           categoriasAtributoQualitativo);
74     if (categoriasAtributoQualitativo.size() > 0)
75     {
76         ui.listWidgetValores->item(0)->setCheckState(Qt::Checked);
77     }
78 }
79
80 void WidgetCategoriasQual::itemChanged(QListWidgetItem *item)
81 {
82     ui.listWidgetValores->blockSignals(true);
83
84     const int count = ui.listWidgetValores->count();
85
86     bool algumMarcado = false;
87
88     for (int i = 0; i < count && !algumMarcado; ++i)
89     {
90         if (ui.listWidgetValores->item(i)->checkState() == Qt::Checked)
91         {
92             algumMarcado = true;
93         }
94     }
95
96     if (!algumMarcado)
97     {
98         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
               fromLatin1("Há de ter pelo menos um item marcado"));
99         item->setCheckState(Qt::Checked);
100    }
101
102    ui.listWidgetValores->blockSignals(false);
103
104 }
105
106 void WidgetCategoriasQual::obterSelecionados(vector<ValorBase *const> &
       selecionados) const
107 {
108     const string atributoSelecionado = ui.comboBoxAtributos->currentText().
           toLatin1().data();
109
110     const int count = ui.listWidgetValores->count();
111
112     for (int i = 0; i < count; ++i)
113     {
114         if (ui.listWidgetValores->item(i)->checkState() == Qt::Checked)
115         {
116             ValorBase *valor = TabelaDeRegistros::obterInstancia()->
                   gerarValor(atributoSelecionado, ui.listWidgetValores->item
                   (i)->text().toLatin1().data());
117             selecionados.push_back(valor);
118         }
119     }
120 }
121
122 bool WidgetCategoriasQual::considerarMissings() const
123 {
124     return ui.checkBoxConsMissings->isChecked();
125 }
```

```
126
127 void WidgetCategoriasQual::setVisibilidadeComboAtributos(const bool visivel
        )
128 {
129     ui.comboBoxAtributos->setVisible(visivel);
130     ui.labelAtributo->setVisible(visivel);
131 }
132
133 void WidgetCategoriasQual::marcarTodos()
134 {
135     const int count = ui.listWidgetValores->count();
136
137     for (int i = 0; i < count; ++i)
138     {
139         ui.listWidgetValores->item(i)->setCheckState(Qt::Checked);
140     }
141 }
142
143 bool WidgetCategoriasQual::haRegistros(string &erro, vector<size_t> &
        registrosValidos) const
144 {
145     vector<ValorBase *const> selecionados = vector<ValorBase *const>();
146     obterSelecionados(selecionados);
147
148     if (selecionados.empty())
149     {
150         erro = "Nenhum valor selecionado";
151         return false;
152     }
153
154     const bool missingsTmb = ui.checkBoxConsMissings->isChecked();
155
156     const string &atributo = _atributos[ui.comboBoxAtributos->currentIndex
            ()];
157     TabelaDeRegistros::obterInstancia()->identificarRegistrosPorValor(
            atributo, selecionados, missingsTmb, false, registrosValidos);
158
159     for (vector<ValorBase *const>::const_iterator citSelecionado =
            selecionados.cbegin();
160         citSelecionado != selecionados.cend();
161         citSelecionado++)
162     {
163         delete *citSelecionado;
164     }
165
166     if (registrosValidos.empty())
167     {
168         erro = "Não foram encontrados registros de acordo com as categorias
                selecionadas";
169         return false;
170     }
171
172     return true;
173 }
174
175 const string &WidgetCategoriasQual::atributoAtual() const
176 {
177     return _atributos[ui.comboBoxAtributos->currentIndex()];
178 }
179
180 void WidgetCategoriasQual::checkConsiderarMissings(bool checked)
181 {
182     ui.checkBoxConsMissings->setChecked(checked);
183 }
```

Listagem B.75: widget_categorias_qual.cpp

```
1  #ifndef WIDGET_CATEGORIAS_QUAL_H
2  #define WIDGET_CATEGORIAS_QUAL_H
3
4  #include "ui_widget_categorias_qual.h"
5
6  #include <string>
7  #include <vector>
8
9  #include "valor_base.h"
10
11 using namespace std;
12
```

```
13 class WidgetCategoriasQual : public QWidget
14 {
15     Q_OBJECT
16
17 public:
18     WidgetCategoriasQual(const vector<string> &atributos, QWidget *parent =
           0);
19     WidgetCategoriasQual(const string &atributo, QWidget *parent = 0);
20     ~WidgetCategoriasQual();
21     void inicialize(const vector<string> &atributos);
22     void setHabilitadoComboBoxAtributos(const bool habilitado);
23     void setAtributoAtualComboBoxAtributos(const string &atributo);
24     void setVisibilidadeComboAtributos(const bool visivel);
25     void obterSelecionados(vector<ValorBase *const> &selecionados) const;
26     bool considerarMissings() const;
27     void marcarTodos();
28     void checkConsiderarMissings(bool checked);
29     bool haRegistros(string &erro, vector<size_t> &registrosValidos) const;
30     const string &atributoAtual() const;
31
32 private:
33     Ui::WidgetCategoriasQual ui;
34     vector<string> _atributos;
35
36 private slots:
37     void currentIndexChangedComboBoxAtributos(const QString&);
38     void itemChanged(QListWidgetItem*);
39 };
40
41 #endif // WIDGET_CATEGORIAS_QUAL_H
```

Listagem B.76: widget_categorias_qual.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>WidgetCategoriasQual</class>
4   <widget class="QWidget" name="WidgetCategoriasQual">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>329</width>
10     <height>234</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>WidgetCategoriasQual</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout">
17    <item>
18     <layout class="QFormLayout" name="formLayout">
19      <item row="0" column="0">
20       <widget class="QLabel" name="labelAtributo">
21        <property name="text">
22         <string>Atributo:</string>
23        </property>
24       </widget>
25      </item>
26      <item row="0" column="1">
27       <widget class="QComboBox" name="comboBoxAtributos"/>
28      </item>
29      <item row="1" column="0">
30       <widget class="QLabel" name="label_3">
31        <property name="text">
32         <string>Categorias:</string>
33        </property>
34       </widget>
35      </item>
36      <item row="1" column="1">
37       <widget class="QListWidget" name="listWidgetValores">
38        <property name="sizePolicy">
39         <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
40          <horstretch>0</horstretch>
41          <verstretch>0</verstretch>
42         </sizepolicy>
43        </property>
44        <property name="maximumSize">
45         <size>
46          <width>16777215</width>
```

```
47            <height>16777215</height>
48           </size>
49          </property>
50         </widget>
51        </item>
52       </layout>
53      </item>
54      <item>
55       <widget class="QGroupBox" name="groupBox">
56        <property name="title">
57         <string>Considerar também:</string>
58        </property>
59        <layout class="QHBoxLayout" name="horizontalLayout_2">
60         <item>
61          <widget class="QCheckBox" name="checkBoxConsMissings">
62           <property name="text">
63            <string>Missings</string>
64           </property>
65          </widget>
66         </item>
67         <item>
68          <spacer name="horizontalSpacer_2">
69           <property name="orientation">
70            <enum>Qt::Horizontal</enum>
71           </property>
72           <property name="sizeHint" stdset="0">
73            <size>
74             <width>40</width>
75             <height>20</height>
76            </size>
77           </property>
78          </spacer>
79         </item>
80        </layout>
81       </widget>
82      </item>
83     </layout>
84    </widget>
85    <layoutdefault spacing="6" margin="11"/>
86    <resources/>
87    <connections/>
88   </ui>
```

Listagem B.77: widget_categorias_qual.ui

```
1  #include "widget_distribuicao.h"
2
3  #include <QMessageBox>
4  #include <QDebug>
5
6  #include "utilidades.h"
7
8  WidgetDistribuicao::WidgetDistribuicao(
9      const list<double> &valores,
10     const Distribuicao &distribuicao,
11     QWidget *parent)
12     : QWidget(parent), _distribuicao(distribuicao), _valores(valores)
13 {
14     ui.setupUi(this);
15
16     const string nome = _distribuicao.nome();
17     const vector<string> parametrosNomes = _distribuicao.nomesParametros();
18     const vector<double> parametrosDbl = _distribuicao.parametros();;
19
20     ui.groupBox->setTitle(QString::fromLatin1("Parâmetros da ") +QString::
           fromLatin1(nome.c_str()));
21
22     setLabelVisivelValorP();
23
24     for (size_t i = 0; i < parametrosNomes.size(); i++)
25     {
26         const string &nomeParametro = parametrosNomes[i];
27         const double valorInicial = parametrosDbl[i];
28
29         WidgetParametro *widgetParametro = new WidgetParametro(
               nomeParametro, valorInicial, ui.widgetParametros);
30         ui.widgetParametros->layout()->addWidget(widgetParametro);
31
32         _widgetsParametros.push_back(widgetParametro);
```

```
33      }
34
35      setLabelVisivelValorP();
36
37      connect(ui.pushButtonGerarDistri, SIGNAL(clicked()), this, SLOT(
            gerarDistribuicaoClick()));
38  }
39
40  WidgetDistribuicao::~WidgetDistribuicao()
41  {
42      const void * address = static_cast<const void*>(&(*this));
43      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
44  }
45
46  const Distribuicao &WidgetDistribuicao::distribuicao() const
47  {
48      return _distribuicao;
49  }
50
51  void WidgetDistribuicao::setLabelVisivelValorP()
52  {
53      const bool visivel = _distribuicao.valorPValido();
54
55      ui.labelValorPedit->setText(QString::fromLatin1(Utilidades::dblToStdStr
            (_distribuicao.valorP(), 15, true).c_str()));
56      ui.labelValorPedit->setVisible(visivel);
57      ui.labelValorP->setVisible(visivel);
58  }
59
60
61  void WidgetDistribuicao::gerarDistribuicaoClick()
62  {
63      Sucesso sucesso = Sucesso::OK;
64
65      vector<double> parametrosInseridos = vector<double>();
66
67      for (vector<WidgetParametro*>::const_iterator citParametroWidget =
            _widgetsParametros.cbegin();
68          citParametroWidget != _widgetsParametros.cend();
69          citParametroWidget++)
70      {
71          parametrosInseridos.push_back((*citParametroWidget)->valorParametro
                ());
72      }
73
74      Distribuicao &antiga = _distribuicao;
75
76      try
77      {
78          _distribuicao = Distribuicao(parametrosInseridos, _valores,
                _distribuicao.tipo());
79      }
80      catch (fit::freq_table_exception &e)
81      {
82          sucesso = Sucesso::ERRO_QUI_QUADRADO;
83      }
84      catch (fit::value_exception &e)
85      {
86          sucesso = Sucesso::ERRO_QUI_QUADRADO;
87      }
88      catch (fit::distribution_exception &e)
89      {
90          sucesso = Sucesso::ERRO_PARAMETRO;
91      }
92      catch (...)
93      {
94          sucesso = Sucesso::ERRO_INDEFINIDO;
95      }
96
97      bool resetarPatametros = false;
98
99      if (sucesso != Sucesso::OK)
100     {
101         switch (sucesso)
102         {
103         case Sucesso::ERRO_QUI_QUADRADO:
104             {
105                 QMessageBox msgQuestion(QMessageBox::Question, "Aviso",
                        QString::fromLatin1("Não foi possível calcular o teste
                         Qui-Quadrado com a distribuição gerada com os
                        parâmetros escolhidos!\nDeseja continuar com está
                        distribuição mesmo assim?"), QMessageBox::Yes|
```

```
                    QMessageBox::No, this);
106                 msgQuestion.button(QMessageBox::Yes)->setText(QString::
                        fromLatin1("Sim"));
107                 msgQuestion.button(QMessageBox::No)->setText(QString::
                        fromLatin1("Não"));
108
109                 if (msgQuestion.exec() == QMessageBox::Yes)
110                 {
111                     _distribuicao = Distribuicao(parametrosInseridos, list<
                            double>(), _distribuicao.tipo());
112                 }
113                 else
114                 {
115                     resetarPatametros = true;
116                 }
117                 break;
118             }
119         case Sucesso::ERRO_PARAMETRO:
120             {
121                 QMessageBox::warning(NULL, QString::fromLatin1("Aviso"),
                        QString::fromLatin1("Não foi possível gerar a
                        distribuição com os parâmetros escolhidos!"));
122                 resetarPatametros = true;
123                 break;
124             }
125         case Sucesso::ERRO_INDEFINIDO:
126             {
127                 QMessageBox::critical(NULL, QString::fromLatin1("Erro"),
                        QString::fromLatin1("Ocorreu um erro indefinido!"));
128                 resetarPatametros = true;
129                 break;
130             }
131         }
132     }
133
134     if (resetarPatametros)
135     {
136         parametrosInseridos = _distribuicao.parametros();
137         vector<double>::const_iterator citParametro = parametrosInseridos.
                cbegin();
138
139         for (vector<WidgetParametro*>::const_iterator citParametroWidget =
                _widgetsParametros.cbegin();
140             citParametroWidget != _widgetsParametros.cend();
141             citParametroWidget++, citParametro++)
142         {
143             (*citParametroWidget)->setValorParametro(*citParametro);
144         }
145     }
146
147     setLabelVisivelValorP();
148 }
```

Listagem B.78: widget_distribuicao.cpp

```
1  #ifndef WIDGET_DISTRIBUICAO_H
2  #define WIDGET_DISTRIBUICAO_H
3
4  #include "ui_widget_distribuicao.h"
5
6  #include "widget_parametro.h"
7
8  #include <list>
9
10 #include "distribuicao_controle.h"
11
12 using namespace std;
13
14 class WidgetDistribuicao : public QWidget
15 {
16     Q_OBJECT
17
18 public:
19     enum Sucesso {OK, ERRO_QUI_QUADRADO, ERRO_PARAMETRO, ERRO_INDEFINIDO};
20
21     WidgetDistribuicao(
22         const list<double> &valores,
23         const Distribuicao &distribuicao,
24         QWidget *parent = 0);
```

```
25      ~WidgetDistribuicao();
26
27      const Distribuicao &distribuicao() const;
28
29  private slots:
30      void gerarDistribuicaoClick();
31
32  private:
33
34      Ui::WidgetDistribuicao ui;
35      vector<WidgetParametro*> _widgetsParametros;
36      Distribuicao _distribuicao;
37      const list<double> &_valores;
38
39      void setLabelVisivelValorP();
40  };
41
42  #endif // WIDGET_DISTRIBUICAO_H
```

Listagem B.79: widget_distribuicao.h

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <ui version="4.0">
3    <class>WidgetDistribuicao</class>
4    <widget class="QWidget" name="WidgetDistribuicao">
5     <property name="geometry">
6      <rect>
7       <x>0</x>
8       <y>0</y>
9       <width>379</width>
10      <height>233</height>
11     </rect>
12    </property>
13    <property name="windowTitle">
14     <string>WidgetDistribuicao</string>
15    </property>
16    <layout class="QVBoxLayout" name="verticalLayout_2">
17     <item>
18      <widget class="QGroupBox" name="groupBox">
19       <property name="title">
20        <string>Parâmetros da ...</string>
21       </property>
22       <layout class="QVBoxLayout" name="verticalLayout">
23        <item>
24         <layout class="QFormLayout" name="formLayout">
25          <item row="0" column="0">
26           <widget class="QLabel" name="labelValorP">
27            <property name="text">
28             <string>Valor-p</string>
29            </property>
30           </widget>
31          </item>
32          <item row="0" column="1">
33           <widget class="QLabel" name="labelValorPedit">
34            <property name="text">
35             <string>TextLabel</string>
36            </property>
37           </widget>
38          </item>
39         </layout>
40        </item>
41        <item>
42         <widget class="QWidget" name="widgetParametros" native="true">
43          <layout class="QVBoxLayout" name="verticalLayout_3">
44           <property name="leftMargin">
45            <number>0</number>
46           </property>
47           <property name="topMargin">
48            <number>0</number>
49           </property>
50           <property name="rightMargin">
51            <number>0</number>
52           </property>
53           <property name="bottomMargin">
54            <number>0</number>
55           </property>
56          </layout>
57         </widget>
58        </item>
```

```
59          <item>
60           <layout class="QHBoxLayout" name="horizontalLayout">
61            <item>
62             <spacer name="horizontalSpacer">
63              <property name="orientation">
64               <enum>Qt::Horizontal</enum>
65              </property>
66              <property name="sizeHint" stdset="0">
67               <size>
68                <width>40</width>
69                <height>20</height>
70               </size>
71              </property>
72             </spacer>
73            </item>
74            <item>
75             <widget class="QPushButton" name="pushButtonGerarDistri">
76              <property name="text">
77               <string>Gerar distribuição com o parametros definidos</string>
78              </property>
79             </widget>
80            </item>
81           </layout>
82          </item>
83          <item>
84           <spacer name="verticalSpacer">
85            <property name="orientation">
86             <enum>Qt::Vertical</enum>
87            </property>
88            <property name="sizeHint" stdset="0">
89             <size>
90              <width>20</width>
91              <height>40</height>
92             </size>
93            </property>
94           </spacer>
95          </item>
96         </layout>
97        </widget>
98       </item>
99      </layout>
100    </widget>
101    <layoutdefault spacing="6" margin="11"/>
102    <resources/>
103    <connections/>
104   </ui>
```

Listagem B.80: widget_distribuicao.ui

```
1  #include "widget_escolher_dp.h"
2
3  #include "gerador_distribuicoes.h"
4  #include "tabela_registros.h"
5  #include "populador.h"
6
7  #include <QMessageBox>
8  #include <QDebug>
9
10 WidgetEscolherDP::WidgetEscolherDP(
11     const vector<string> &atributos,
12     const vector<vector<size_t>> &indicesValidosPorAtributo,
13     const WidgetEscolherDP::Opcao opcao,
14     QWidget *parent)
15     : QWidget(parent)
16 {
17     inicialize(atributos, indicesValidosPorAtributo, opcao);
18
19     if (parent)
20     {
21         connect(ui.pushButtonCancelar, SIGNAL(clicked()), parent, SLOT(
                reject()));
22     }
23 }
24
25 WidgetEscolherDP::WidgetEscolherDP(
26     const string &atributo,
27     const vector<size_t> &indicesValidos,
28     const WidgetEscolherDP::Opcao opcao,
29     QWidget *parent)
```

```
30      : QWidget(parent)
31  {
32      vector<string> atributos = vector<string>();
33      atributos.push_back(atributo);
34
35      vector<vector<size_t>> indicesValidosPorAtributo = vector<vector<size_t
            >>();
36      indicesValidosPorAtributo.push_back(indicesValidos);
37
38      inicialize(atributos, indicesValidosPorAtributo, opcao);
39
40      if (parent)
41      {
42          connect(ui.pushButtonCancelar, SIGNAL(clicked()), parent, SLOT(
                reject()));
43      }
44  }
45
46  WidgetEscolherDP::~WidgetEscolherDP()
47  {
48      const void * address = static_cast<const void*>(&(*this));
49      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
50  }
51
52  void WidgetEscolherDP::inicialize(const vector<string> &atributos, const
        vector<vector<size_t>> &indicesValidosPorAtributo, const
        WidgetEscolherDP::Opcao opcao)
53  {
54      ui.setupUi(this);
55
56      _atributos = atributos;
57
58      ui.pushButtonGerar->setVisible(opcao == ESCOLHER);
59      ui.pushButtonCancelar->setVisible(opcao == ESCOLHER);
60      if (opcao != ESCOLHER)
61      {
62          ui.horizontalSpacer->changeSize(0,0, QSizePolicy::Fixed,
                QSizePolicy::Fixed);
63      }
64
65      vector<vector<double>> valoresDblPorAtributo = vector<vector<double>>()
            ;
66      TabelaDeRegistros::obterInstancia()->
            valoresAtributosQuantDblPorIndicePorAtributo(atributos,
            indicesValidosPorAtributo, valoresDblPorAtributo);
67
68      _valoresListPorAtributo = vector<list<double>>();
69      for (vector<vector<double>>::const_iterator citColunaVec =
            valoresDblPorAtributo.cbegin();
70          citColunaVec != valoresDblPorAtributo.cend();
71          citColunaVec++)
72      {
73          _valoresListPorAtributo.push_back(list<double>(citColunaVec->cbegin
                (), citColunaVec->cend()));
74          list<double> &colunaList = _valoresListPorAtributo.back();
75          colunaList.sort();
76      }
77
78      setupTabs();
79      ui.stackedWidget->setCurrentIndex(0);
80
81      Populador::populeComboBox(ui.comboBoxAtributo, _atributos);
82      ui.comboBoxAtributo->setCurrentIndex(0);
83      connect(ui.comboBoxAtributo, SIGNAL(currentIndexChanged(int)), ui.
            stackedWidget, SLOT(setCurrentIndex(int)));
84
85      if (_atributos.size() == 1)
86      {
87          ui.comboBoxAtributo->setEnabled(false);
88          //ui.labelAtributo->setVisible(false);
89          //ui.horizontalSpacer_2->changeSize(0, 0, QSizePolicy::Fixed,
                QSizePolicy::Fixed);
90      }
91
92      connect(ui.pushButtonGerar, SIGNAL(clicked()), this, SLOT(gerarClick())
            );
93      connect(ui.pushButtonCancelar, SIGNAL(clicked()), this, SLOT(reject()))
            ;
94  }
95
96  const Distribuicao &WidgetEscolherDP::distribuicaoAtual() const
97  {
```

```
98      QTabWidget *atributoTabs = dynamic_cast<QTabWidget*>(ui.stackedWidget->
            currentWidget());
99      WidgetDistribuicao *widgetDistribuicaoAtual = dynamic_cast<
            WidgetDistribuicao*>(atributoTabs->currentWidget());
100
101     return widgetDistribuicaoAtual->distribuicao();
102 }
103
104 void WidgetEscolherDP::setupTabs()
105 {
106     vector<list<double>>::const_iterator citColuna =
            _valoresListPorAtributo.cbegin();
107
108     for (vector<string>::const_iterator citAtributo = _atributos.cbegin();
109          citAtributo != _atributos.cend();
110          citAtributo++, citColuna++)
111     {
112         QTabWidget *tabs = new QTabWidget(this);
113         ui.stackedWidget->addWidget(tabs);
114
115         const string &atributo = *citAtributo;
116         const list<double> &valores = *citColuna;
117
118         vector<const Distribuicao> distribuicoesOk = vector<const
                Distribuicao>();
119         vector<const Distribuicao> distribuicoesNaoOk = vector<const
                Distribuicao>();
120         GeradorDistribuicoes::gerarTodasDistribuicoes(valores,
                distribuicoesOk, distribuicoesNaoOk);
121
122         size_t iTab = 0;
123         for (vector<const Distribuicao>::const_reverse_iterator
                citDistribuicao = distribuicoesOk.crbegin();
124              citDistribuicao != distribuicoesOk.rend();
125              citDistribuicao++, iTab++)
126         {
127             WidgetDistribuicao *widgetDistribuicao = new WidgetDistribuicao
                    (valores, *citDistribuicao, this);
128             _widgetsDistribuicoes.push_back(widgetDistribuicao);
129
130             tabs->insertTab(iTab, widgetDistribuicao, QString::fromLatin1(
                    citDistribuicao->nome().c_str()));
131         }
132
133         string msg = "Atributo: " +atributo +"\n\nNão foi possível
                construir a partir dos dados as seguintes distribuições de
                probabilidade:\n";
134         for (vector<const Distribuicao>::const_iterator
                citDistribuicaoNaoOk = distribuicoesNaoOk.cbegin();
135              citDistribuicaoNaoOk != distribuicoesNaoOk.cend();
136              citDistribuicaoNaoOk++)
137         {
138             WidgetDistribuicao *widgetDistribuicao = new WidgetDistribuicao
                    (valores, *citDistribuicaoNaoOk, this);
139
140             msg += "\t- " +widgetDistribuicao->distribuicao().nome() +"\n";
141
142             _widgetsDistribuicoes.push_back(widgetDistribuicao);
143
144             tabs->insertTab(iTab, widgetDistribuicao, QString::fromLatin1(
                    citDistribuicaoNaoOk->nome().c_str()));
145         }
146         msg += "\nMesmo assim você pode utilizá-las definindo seus
                parâmetros!";
147
148         if (!distribuicoesNaoOk.empty())
149         {
150             QMessageBox::warning(NULL, QString::fromLatin1("Aviso"),
                    QString::fromLatin1(msg.c_str()));
151         }
152     }
153 }
154
155 void WidgetEscolherDP::gerarClick()
156 {
157     vector<const Distribuicao> distribuicoes = vector<const Distribuicao>()
            ;
158
159     for (size_t i = 0; i < ui.stackedWidget->count(); i++)
160     {
161         QTabWidget *atributoTabs = dynamic_cast<QTabWidget*>(ui.
                stackedWidget->widget(i));
```

```
162             WidgetDistribuicao *distriWidgetAtual = dynamic_cast<
                    WidgetDistribuicao*>(atributoTabs->currentWidget());
163
164             const Distribuicao &distribuicao = distriWidgetAtual->distribuicao
                    ();
165             distribuicoes.push_back(distribuicao);
166         }
167
168         emit distribuicoesEscolhidas(_atributos, distribuicoes);
169     }
```

Listagem B.81: widget_escolher_dp.cpp

```
1  #ifndef WIDGET_ESCOLHER_DP_H
2  #define WIDGET_ESCOLHER_DP_H
3
4  #include <QWidget>
5  #include "ui_widget_escolher_dp.h"
6
7  #include "widget_distribuicao.h"
8
9  #include <vector>
10 #include <utility>
11
12 using namespace std;
13
14 class WidgetEscolherDP : public QWidget
15 {
16     Q_OBJECT
17
18 public:
19     enum Opcao {NADA, ESCOLHER};
20
21     WidgetEscolherDP(
22         const vector<string> &atributos,
23         const vector<vector<size_t>> &indicesValidosPorAtributo,
24         const WidgetEscolherDP::Opcao opcao,
25         QWidget *parent = 0);
26     WidgetEscolherDP(
27         const string &atributo,
28         const vector<size_t> &indicesValidos,
29         const WidgetEscolherDP::Opcao opcao,
30         QWidget *parent = 0);
31
32     ~WidgetEscolherDP();
33
34     const Distribuicao &distribuicaoAtual() const;
35
36 private:
37     Ui::WidgetEscolherDP ui;
38
39     vector<WidgetDistribuicao*> _widgetsDistribuicoes;
40     vector<string> _atributos;
41     vector<vector<size_t>> _indicesValidosPorAtributo;
42     vector<list<double>> _valoresListPorAtributo;
43
44     void inicialize(
45         const vector<string> &atributos,
46         const vector<vector<size_t>> &indicesValidosPorAtributo,
47         const WidgetEscolherDP::Opcao opcao);
48
49     void setupTabs();
50
51 signals:
52     void distribuicoesEscolhidas(const vector<string>&, const vector<const
          Distribuicao>&);
53
54 private slots:
55     void gerarClick();
56 };
57
58 #endif // WIDGET_ESCOLHER_DP_H
```

Listagem B.82: widget_escolher_dp.h

```
1 <?xml version="1.0" encoding="UTF-8"?>
```

```
 2 <ui version="4.0">
 3  <class>WidgetEscolherDP</class>
 4  <widget class="QWidget" name="WidgetEscolherDP">
 5   <property name="geometry">
 6    <rect>
 7     <x>0</x>
 8     <y>0</y>
 9     <width>595</width>
10     <height>399</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>WidgetEscolherDP</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout">
17    <property name="leftMargin">
18     <number>0</number>
19    </property>
20    <property name="topMargin">
21     <number>0</number>
22    </property>
23    <property name="rightMargin">
24     <number>0</number>
25    </property>
26    <property name="bottomMargin">
27     <number>0</number>
28    </property>
29    <item>
30     <layout class="QHBoxLayout" name="horizontalLayout_2">
31      <item>
32       <widget class="QLabel" name="labelAtributo">
33        <property name="text">
34         <string>Atributo</string>
35        </property>
36       </widget>
37      </item>
38      <item>
39       <widget class="QComboBox" name="comboBoxAtributo"/>
40      </item>
41      <item>
42       <spacer name="horizontalSpacer_2">
43        <property name="orientation">
44         <enum>Qt::Horizontal</enum>
45        </property>
46        <property name="sizeHint" stdset="0">
47         <size>
48          <width>40</width>
49          <height>20</height>
50         </size>
51        </property>
52       </spacer>
53      </item>
54     </layout>
55    </item>
56    <item>
57     <widget class="QLabel" name="labelAviso">
58      <property name="text">
59       <string>Por favor, para cada atributo selecione a aba correspondete a
              distribuição desejada.</string>
60      </property>
61     </widget>
62    </item>
63    <item>
64     <widget class="QStackedWidget" name="stackedWidget"/>
65    </item>
66    <item>
67     <layout class="QHBoxLayout" name="horizontalLayout">
68      <item>
69       <spacer name="horizontalSpacer">
70        <property name="orientation">
71         <enum>Qt::Horizontal</enum>
72        </property>
73        <property name="sizeHint" stdset="0">
74         <size>
75          <width>40</width>
76          <height>20</height>
77         </size>
78        </property>
79       </spacer>
80      </item>
81      <item>
82       <widget class="QPushButton" name="pushButtonGerar">
```

```
83          <property name="text">
84           <string>Gerar</string>
85          </property>
86         </widget>
87        </item>
88        <item>
89         <widget class="QPushButton" name="pushButtonCancelar">
90          <property name="text">
91           <string>Cancelar</string>
92          </property>
93         </widget>
94        </item>
95       </layout>
96      </item>
97     </layout>
98    </widget>
99    <layoutdefault spacing="6" margin="11"/>
100   <resources/>
101   <connections/>
102  </ui>
```

Listagem B.83: widget_escolher_dp.ui

```
1   #include "widget_intervalo_quant.h"
2
3   #include <QDebug>
4
5   WidgetIntervaloQuant::WidgetIntervaloQuant(
6       const double max,
7       const double min,
8       QWidget *parent)
9       : QWidget(parent)
10  {
11      ui.setupUi(this);
12
13      connect(ui.pushButtonRemover, SIGNAL(clicked()), this, SLOT(
            removerClick()));
14      connect(ui.comboBoxOpInf, SIGNAL(currentIndexChanged(const QString&)),
            this, SLOT(currentIndexChangedcomboBoxOpInf(const QString&)));
15
16      ui.comboBoxOpInf->addItem("<=");
17      ui.comboBoxOpInf->addItem("<");
18      ui.comboBoxOpInf->addItem("=");
19
20      ui.comboBoxOpSup->addItem("<=");
21      ui.comboBoxOpSup->addItem("<");
22
23      ui.doubleSpinBoxSup->setMaximum(max);
24      ui.doubleSpinBoxSup->setValue(max);
25      ui.doubleSpinBoxInf->setMinimum(min);
26      ui.doubleSpinBoxInf->setValue(min);
27  }
28
29  WidgetIntervaloQuant::~WidgetIntervaloQuant()
30  {
31      const void * address = static_cast<const void*>(&(*this));
32      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
33  }
34
35
36  TabelaDeRegistros::Intervalo WidgetIntervaloQuant::gerarIntervalo() const
37  {
38      Operacao op;
39
40      double inferior = ui.doubleSpinBoxInf->value();
41      double superior = ui.doubleSpinBoxSup->value();
42
43      if (ui.comboBoxOpInf->currentText() == "<" && ui.comboBoxOpSup->
            currentText() == "<")
44      {
45          op = MAIORQUE_MENORQUE;
46      }
47      else if (ui.comboBoxOpInf->currentText() == "<" && ui.comboBoxOpSup->
            currentText() == "<=")
48      {
49          op = MAIORQUE_MENORIGUALQUE;
50      }
51      else if (ui.comboBoxOpInf->currentText() == "<=" && ui.comboBoxOpSup->
            currentText() == "<")
```

```
52      {
53          op = MAIORIGUALQUE_MENORQUE;
54      }
55      else if (ui.comboBoxOpInf->currentText() == "<=" && ui.comboBoxOpSup->
            currentText() == "<=")
56      {
57          op = MAIORIGUALQUE_MENORIGUALQUE;
58      }
59      else if (ui.comboBoxOpInf->currentText() == "=")
60      {
61          op = MAIORIGUALQUE_MENORIGUALQUE;
62          superior = inferior;
63      }
64
65      if (superior < inferior)
66      {
67          throw exception("Um dos intervalos tem valor superior menor que o
                inferior!");
68      }
69
70      return TabelaDeRegistros::Intervalo(inferior, superior, op);
71  }
72
73  void WidgetIntervaloQuant::removerClick()
74  {
75      emit removerIntervaloWidget(this);
76  }
77
78  void WidgetIntervaloQuant::currentIndexChangedcomboBoxOpInf(const QString &
        currentText)
79  {
80      bool igual = currentText == "=";
81
82      ui.comboBoxOpSup->setEnabled(!igual);
83      ui.doubleSpinBoxSup->setEnabled(!igual);
84  }
```

Listagem B.84: widget_intervalo_quant.cpp

```
1  #ifndef WIDGET_INTERVALO_QUANT_H
2  #define WIDGET_INTERVALO_QUANT_H
3
4  #include "ui_widget_intervalo_quant.h"
5
6  #include "tabela_registros.h"
7
8  using namespace std;
9
10 class WidgetIntervaloQuant : public QWidget
11 {
12     Q_OBJECT
13
14 public:
15     WidgetIntervaloQuant(
16         const double max,
17         const double min,
18         QWidget *parent = 0);
19     ~WidgetIntervaloQuant();
20     TabelaDeRegistros::Intervalo gerarIntervalo() const;
21
22
23 private:
24     Ui::WidgetIntervaloQuant ui;
25     size_t _indice;
26
27 private slots:
28     void removerClick();
29     void currentIndexChangedcomboBoxOpInf(const QString &currentText);
30
31 signals:
32     void removerIntervaloWidget(WidgetIntervaloQuant*);
33 };
34
35 #endif // T_IntervaloQuantWidget_H
```

Listagem B.85: widget_intervalo_quant.h

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>WidgetIntervaloQuant</class>
 <widget class="QWidget" name="WidgetIntervaloQuant">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>606</width>
    <height>23</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>IntervaloQuant</string>
  </property>
  <layout class="QHBoxLayout" name="horizontalLayout">
   <property name="spacing">
    <number>3</number>
   </property>
   <property name="leftMargin">
    <number>0</number>
   </property>
   <property name="topMargin">
    <number>0</number>
   </property>
   <property name="rightMargin">
    <number>0</number>
   </property>
   <property name="bottomMargin">
    <number>0</number>
   </property>
   <item>
    <spacer name="horizontalSpacer">
     <property name="orientation">
      <enum>Qt::Horizontal</enum>
     </property>
     <property name="sizeHint" stdset="0">
      <size>
       <width>181</width>
       <height>20</height>
      </size>
     </property>
    </spacer>
   </item>
   <item>
    <widget class="QDoubleSpinBox" name="doubleSpinBoxInf">
     <property name="sizePolicy">
      <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
       <horstretch>0</horstretch>
       <verstretch>0</verstretch>
      </sizepolicy>
     </property>
     <property name="minimumSize">
      <size>
       <width>100</width>
       <height>0</height>
      </size>
     </property>
     <property name="alignment">
      <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
     </property>
     <property name="decimals">
      <number>3</number>
     </property>
     <property name="minimum">
      <double>-999999999.000000000000000</double>
     </property>
     <property name="maximum">
      <double>999999999.000000000000000</double>
     </property>
    </widget>
   </item>
   <item>
    <widget class="QComboBox" name="comboBoxOpInf">
     <property name="sizePolicy">
      <sizepolicy hsizetype="Minimum" vsizetype="Fixed">
       <horstretch>0</horstretch>
       <verstretch>0</verstretch>
      </sizepolicy>
     </property>
    </widget>
```

```
82      </item>
83      <item>
84       <widget class="QLabel" name="labelValor">
85        <property name="text">
86         <string>Valor</string>
87        </property>
88        <property name="alignment">
89         <set>Qt::AlignCenter</set>
90        </property>
91       </widget>
92      </item>
93      <item>
94       <widget class="QComboBox" name="comboBoxOpSup">
95        <property name="sizePolicy">
96         <sizepolicy hsizetype="Minimum" vsizetype="Fixed">
97          <horstretch>0</horstretch>
98          <verstretch>0</verstretch>
99         </sizepolicy>
100       </property>
101      </widget>
102     </item>
103     <item>
104      <widget class="QDoubleSpinBox" name="doubleSpinBoxSup">
105       <property name="sizePolicy">
106        <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
107         <horstretch>0</horstretch>
108         <verstretch>0</verstretch>
109        </sizepolicy>
110       </property>
111       <property name="minimumSize">
112        <size>
113         <width>100</width>
114         <height>0</height>
115        </size>
116       </property>
117       <property name="decimals">
118        <number>3</number>
119       </property>
120       <property name="minimum">
121        <double>-999999999.000000000000000</double>
122       </property>
123       <property name="maximum">
124        <double>999999999.000000000000000</double>
125       </property>
126      </widget>
127     </item>
128     <item>
129      <widget class="QPushButton" name="pushButtonRemover">
130       <property name="text">
131        <string>-</string>
132       </property>
133      </widget>
134     </item>
135    </layout>
136   </widget>
137   <layoutdefault spacing="6" margin="11"/>
138   <resources/>
139   <connections/>
140  </ui>
```

Listagem B.86: widget_intervalo_quant.ui

```
1  #include "widget_intervalos_quant.h"
2
3  #include "coletor_estatisticas.h"
4  #include "populador.h"
5
6  #include <algorithm>
7
8  #include <QMessageBox>
9  #include <QDebug>
10
11 WidgetIntervalosQuant::WidgetIntervalosQuant(
12     const vector<string> &atributos,
13     QWidget *parent)
14     : QWidget(parent)
15 {
16     inicialize(atributos);
17 }
```

```
18
19 WidgetIntervalosQuant::WidgetIntervalosQuant(
20     const string &atributo,
21     QWidget *parent)
22     : QWidget(parent)
23 {
24     vector<string> atributos = vector<string>();
25     atributos.push_back(atributo);
26
27     inicialize(atributos);
28 }
29
30 WidgetIntervalosQuant::~WidgetIntervalosQuant()
31 {
32     const void * address = static_cast<const void*>(&(*this));
33     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
34
35     for (vector<WidgetIntervaloQuant*>::const_iterator citIntervalo =
           _widgetsIntervalosQuant.cbegin();
36          citIntervalo != _widgetsIntervalosQuant.cend();
37          citIntervalo++)
38     {
39         delete *citIntervalo;
40     }
41 }
42
43 void WidgetIntervalosQuant::inicialize(const vector<string> &atributos)
44 {
45     ui.setupUi(this);
46
47     _atributos = atributos;
48
49     connect(ui.pushButtonAddIntervalo, SIGNAL(clicked()), this, SLOT(
           addIntervaloClick()));
50     connect(ui.comboBoxAtributosFiltro, SIGNAL(currentIndexChanged(const
           QString&)), this, SLOT(currentIndexChangedComboBoxAtributosFiltro(
           const QString&)));
51
52     Populador::populeComboBox(ui.comboBoxAtributosFiltro, _atributos);
53
54     addIntervaloClick();
55 }
56
57 void WidgetIntervalosQuant::setHabilitadoComboBoxAtributos(const bool
       habilitado)
58 {
59     ui.comboBoxAtributosFiltro->setEnabled(habilitado);
60 }
61
62 void WidgetIntervalosQuant::setAtributoAtualComboBoxAtributos(const string
      &atributo)
63 {
64     vector<string>::const_iterator citAtributo = std::find(_atributos.
           cbegin(), _atributos.cend(), atributo);
65
66     if (citAtributo != _atributos.cend())
67     {
68         ui.comboBoxAtributosFiltro->setCurrentIndex(citAtributo -
               _atributos.cbegin());
69     }
70 }
71
72 void WidgetIntervalosQuant::gerarIntervalos(vector<TabelaDeRegistros::
       Intervalo> &intervalos) const
73 {
74     for (vector<WidgetIntervaloQuant*>::const_iterator citIntervalo =
           _widgetsIntervalosQuant.cbegin();
75          citIntervalo != _widgetsIntervalosQuant.cend();
76          citIntervalo++)
77     {
78         intervalos.push_back((*citIntervalo)->gerarIntervalo());
79     }
80 }
81
82 void WidgetIntervalosQuant::addIntervaloClick()
83 {
84     const string atributo = ui.comboBoxAtributosFiltro->currentText().
           toLatin1().data();
85
86     const ColetorEstatisticas::EstatisticaQuant &estatisticaQuant =
           ColetorEstatisticas::obterInstancia()->
           estatisticaAtributoQuantitativo(atributo);
```

```
87
88      WidgetIntervaloQuant *widgetIntervaloQuant = new WidgetIntervaloQuant(
            estatisticaQuant.maximo + 1, estatisticaQuant.minimo - 1, ui.
            scrollArea);
89      connect(widgetIntervaloQuant, SIGNAL(removerIntervaloWidget(
            WidgetIntervaloQuant*)), this, SLOT(removerIntervaloWidget(
            WidgetIntervaloQuant*)));
90      _widgetsIntervalosQuant.push_back(widgetIntervaloQuant);
91      ui.scrollArea->widget()->layout()->addWidget(widgetIntervaloQuant);
92  }
93
94  void WidgetIntervalosQuant::removerIntervaloWidget(WidgetIntervaloQuant *
        intervaloWidget)
95  {
96      if (_widgetsIntervalosQuant.size() == 1)
97      {
98          QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("É necessário ter pelo menos um intervalo!"));
99          return;
100     }
101
102     vector<WidgetIntervaloQuant*>::const_iterator citIntervalo = std::find(
            _widgetsIntervalosQuant.cbegin(), _widgetsIntervalosQuant.cend(),
            intervaloWidget);
103
104     if (citIntervalo != _widgetsIntervalosQuant.cend())
105     {
106         ui.scrollArea->widget()->layout()->removeWidget(*citIntervalo);
107         delete *citIntervalo;
108         _widgetsIntervalosQuant.erase(citIntervalo);
109     }
110 }
111
112 void WidgetIntervalosQuant::setVisibilidadeComboAtributos(const bool
        visivel)
113 {
114     ui.comboBoxAtributosFiltro->setVisible(visivel);
115     ui.labelAtributoFiltro->setVisible(visivel);
116 }
117
118 void WidgetIntervalosQuant::currentIndexChangedComboBoxAtributosFiltro(
        const QString &text)
119 {
120     const string atributo = text.toLatin1().data();
121
122     if (!TabelaDeRegistros::obterInstancia()->existeAtributo(atributo))
123     {
124         return;
125     }
126
127     for (vector<WidgetIntervaloQuant*>::const_iterator citIntervalo =
            _widgetsIntervalosQuant.cbegin();
128         citIntervalo != _widgetsIntervalosQuant.cend();
129         citIntervalo++)
130     {
131         ui.scrollArea->widget()->layout()->removeWidget(*citIntervalo);
132         delete *citIntervalo;
133     }
134
135     _widgetsIntervalosQuant.clear();
136
137     addIntervaloClick();
138 }
139
140 bool WidgetIntervalosQuant::haRegistro(string &erro, vector<size_t> &
        registrosValidos) const
141 {
142     const bool missingsTmb = ui.checkBoxConsMissings->isChecked();
143     const bool invalidosTmb = ui.checkBoxConsInvalidos->isChecked();
144
145     vector<TabelaDeRegistros::Intervalo> intervalos = vector<
            TabelaDeRegistros::Intervalo>();
146
147     try
148     {
149         gerarIntervalos(intervalos);
150     }
151     catch (std::exception &e)
152     {
153         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1(e.what()));
154         erro = "Erro ao gerar o intervalo!";
```

```
155             return false;
156         }
157
158         const string &atributo = _atributos[ui.comboBoxAtributosFiltro->
                currentIndex()];
159         TabelaDeRegistros::obterInstancia()->
                identificarRegistrosQuantPorIntervalo(atributo, intervalos,
                missingsTmb, invalidosTmb, registrosValidos);
160
161         if (registrosValidos.empty())
162         {
163             erro = "Não foram encontrados registros de acordo com os intervalos
                    inseridos";
164             return false;
165         }
166
167         return true;
168     }
169
170     const string &WidgetIntervalosQuant::atributoAtual() const
171     {
172         return _atributos[ui.comboBoxAtributosFiltro->currentIndex()];
173     }
174
175     bool WidgetIntervalosQuant::considerarMissings() const
176     {
177         return ui.checkBoxConsMissings->isChecked();
178     }
179
180     bool WidgetIntervalosQuant::considerarInvalidos() const
181     {
182         return ui.checkBoxConsInvalidos->isChecked();
183     }
184
185     void WidgetIntervalosQuant::checkConsiderarMissings(bool checked)
186     {
187         ui.checkBoxConsMissings->setChecked(checked);
188     }
189
190     void WidgetIntervalosQuant::checkConsiderarInvalidos(bool checked)
191     {
192         ui.checkBoxConsInvalidos->setChecked(checked);
193     }
```

Listagem B.87: widget_intervalos_quant.cpp

```
1   #ifndef WIDGET_INTERVALOS_QUANT_H
2   #define WIDGET_INTERVALOS_QUANT_H
3
4   #include "ui_widget_intervalos_quant.h"
5
6   #include "widget_intervalo_quant.h"
7
8   #include <vector>
9   #include <string>
10
11  using namespace std;
12
13  class WidgetIntervalosQuant : public QWidget
14  {
15      Q_OBJECT
16
17  public:
18      WidgetIntervalosQuant(const vector<string> &atributos, QWidget *parent
            = 0);
19      WidgetIntervalosQuant(const string &atributo, QWidget *parent = 0);
20      ~WidgetIntervalosQuant();
21      void inicialize(const vector<string> &atributos);
22      void setHabilitadoComboBoxAtributos(const bool habilitado);
23      void setAtributoAtualComboBoxAtributos(const string &atributo);
24      void setVisibilidadeComboAtributos(const bool visivel);
25      void gerarIntervalos(vector<TabelaDeRegistros::Intervalo> &intervalos)
            const;
26      bool haRegistro(string &erro, vector<size_t> &registrosValidos) const;
27      bool considerarMissings() const;
28      bool considerarInvalidos() const;
29      void checkConsiderarMissings(bool checked);
30      void checkConsiderarInvalidos(bool checked);
31      const string &atributoAtual() const;
```

```
32
33 private:
34     Ui::WidgetIntervalosQuant ui;
35     vector<string> _atributos;
36     vector<WidgetIntervaloQuant*> _widgetsIntervalosQuant;
37
38 private slots:
39     void addIntervaloClick();
40     void currentIndexChangedComboBoxAtributosFiltro(const QString&);
41
42 public slots:
43     void removerIntervaloWidget(WidgetIntervaloQuant *intervaloWidget);
44 };
45
46 #endif // WIDGET_INTERVALOS_QUANT_H
```

Listagem B.88: widget_intervalos_quant.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>WidgetIntervalosQuant</class>
4   <widget class="QWidget" name="WidgetIntervalosQuant">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>512</width>
10     <height>374</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>IntervalosQuant</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout">
17    <item>
18     <layout class="QHBoxLayout" name="horizontalLayout">
19      <item>
20       <widget class="QLabel" name="labelAtributoFiltro">
21        <property name="text">
22         <string>Atributo Filtro:</string>
23        </property>
24       </widget>
25      </item>
26      <item>
27       <widget class="QComboBox" name="comboBoxAtributosFiltro">
28        <property name="minimumSize">
29         <size>
30          <width>100</width>
31          <height>0</height>
32         </size>
33        </property>
34       </widget>
35      </item>
36      <item>
37       <spacer name="horizontalSpacer">
38        <property name="orientation">
39         <enum>Qt::Horizontal</enum>
40        </property>
41        <property name="sizeHint" stdset="0">
42         <size>
43          <width>40</width>
44          <height>20</height>
45         </size>
46        </property>
47       </spacer>
48      </item>
49      <item>
50       <widget class="QPushButton" name="pushButtonAddIntervalo">
51        <property name="text">
52         <string>+ Intervalo</string>
53        </property>
54       </widget>
55      </item>
56     </layout>
57    </item>
58    <item>
59     <widget class="QScrollArea" name="scrollArea">
60      <property name="widgetResizable">
61       <bool>true</bool>
```

```
62        </property>
63        <widget class="QWidget" name="scrollAreaWidgetContents">
64         <property name="geometry">
65          <rect>
66           <x>0</x>
67           <y>0</y>
68           <width>492</width>
69           <height>264</height>
70          </rect>
71         </property>
72         <layout class="QVBoxLayout" name="verticalLayout_2">
73          <property name="spacing">
74           <number>2</number>
75          </property>
76          <property name="leftMargin">
77           <number>3</number>
78          </property>
79          <property name="topMargin">
80           <number>3</number>
81          </property>
82          <property name="rightMargin">
83           <number>3</number>
84          </property>
85          <property name="bottomMargin">
86           <number>3</number>
87          </property>
88         </layout>
89        </widget>
90       </widget>
91      </item>
92      <item>
93       <widget class="QGroupBox" name="groupBox">
94        <property name="title">
95         <string>Considerar também:</string>
96        </property>
97        <layout class="QHBoxLayout" name="horizontalLayout_2">
98         <item>
99          <widget class="QCheckBox" name="checkBoxConsMissings">
100          <property name="text">
101           <string>Missings</string>
102          </property>
103         </widget>
104        </item>
105        <item>
106         <spacer name="horizontalSpacer_3">
107          <property name="orientation">
108           <enum>Qt::Horizontal</enum>
109          </property>
110          <property name="sizeType">
111           <enum>QSizePolicy::Fixed</enum>
112          </property>
113          <property name="sizeHint" stdset="0">
114           <size>
115            <width>20</width>
116            <height>20</height>
117           </size>
118          </property>
119         </spacer>
120        </item>
121        <item>
122         <widget class="QCheckBox" name="checkBoxConsInvalidos">
123          <property name="text">
124           <string>Inválidos</string>
125          </property>
126         </widget>
127        </item>
128        <item>
129         <spacer name="horizontalSpacer_2">
130          <property name="orientation">
131           <enum>Qt::Horizontal</enum>
132          </property>
133          <property name="sizeHint" stdset="0">
134           <size>
135            <width>40</width>
136            <height>20</height>
137           </size>
138          </property>
139         </spacer>
140        </item>
141       </layout>
142      </widget>
143     </item>
```

```
144   </layout>
145  </widget>
146  <layoutdefault spacing="6" margin="11"/>
147  <resources/>
148  <connections/>
149 </ui>
```

Listagem B.89: widget_intervalos_quant.ui

```
1  #include "widget_param_diag_caixa.h"
2
3  #include "coletor_estatisticas.h"
4  #include "utilidades.h"
5
6  #include <QDebug>
7
8  WidgetParamDiagCaixa::WidgetParamDiagCaixa(const bool telaImportacao, const
       string &atributo, QWidget *parent)
9      : QWidget(parent), _atributo(atributo)
10 {
11     ui.setupUi(this);
12
13     inicilizarLabels();
14
15     ui.doubleSpinBoxFatorWS->setValue(ColetorEstatisticas::
           _FATOR_WISKER_SUPERIOR);
16     ui.doubleSpinBoxFatorWI->setValue(ColetorEstatisticas::
           _FATOR_WISKER_INFEIROR);
17
18     connect(ui.pushButtonAplicarTodos, SIGNAL(clicked()), this, SLOT(
           emitirParametros()));
19
20     if (!telaImportacao)
21     {
22         ui.pushButtonAplicarTodos->setVisible(false);
23         ui.horizontalSpacer->changeSize(0,0, QSizePolicy::Fixed,
               QSizePolicy::Fixed);
24     }
25 }
26
27 WidgetParamDiagCaixa::~WidgetParamDiagCaixa()
28 {
29     const void * address = static_cast<const void*>(&(*this));
30     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
31 }
32
33 void WidgetParamDiagCaixa::inicilizarLabels()
34 {
35     const ColetorEstatisticas::EstatisticaQuant &estatistica =
           ColetorEstatisticas::obterInstancia()->
           estatisticaAtributoQuantitativo(_atributo);
36
37     ui.labelDI->setText(QString::fromLatin1(Utilidades::dblToStdStr(
           estatistica.desvioInterquartilico, 3).c_str()));
38     ui.labelQS->setText(QString::fromLatin1(Utilidades::dblToStdStr(
           estatistica.quartilSuperior, 3).c_str()));
39     ui.labelQI->setText(QString::fromLatin1(Utilidades::dblToStdStr(
           estatistica.quartilInferior, 3).c_str()));
40 }
41
42 double WidgetParamDiagCaixa::fatorWS() const
43 {
44     return ui.doubleSpinBoxFatorWS->value();
45 }
46
47 double WidgetParamDiagCaixa::fatorWI() const
48 {
49     return ui.doubleSpinBoxFatorWI->value();
50 }
51
52 void WidgetParamDiagCaixa::setFatorWI(const double fatorWI)
53 {
54     ui.doubleSpinBoxFatorWI->setValue(fatorWI);
55 }
56
57 void WidgetParamDiagCaixa::setFatorWS(const double fatorWS)
58 {
59     ui.doubleSpinBoxFatorWS->setValue(fatorWS);
60 }
```

```
61
62 void WidgetParamDiagCaixa::emitirParametros()
63 {
64     emit aplicarParamsDCTodos(fatorWI(), fatorWS());
65 }
```

Listagem B.90: widget_param_diag_caixa.cpp

```
1  #ifndef WIDGET_PARAM_DIAG_CAIXA_H
2  #define WIDGET_PARAM_DIAG_CAIXA_H
3
4  #include "ui_widget_param_diag_caixa.h"
5
6  #include <string>
7
8  using namespace std;
9
10 class WidgetParamDiagCaixa : public QWidget
11 {
12     Q_OBJECT
13
14 public:
15     WidgetParamDiagCaixa(const bool telaImportacao, const string &atributo,
           QWidget *parent = 0);
16     ~WidgetParamDiagCaixa();
17     double fatorWS() const;
18     double fatorWI() const;
19     void setFatorWI(const double fatorWI);
20     void setFatorWS(const double fatorWS);
21
22 private:
23     Ui::WidgetParamDiagCaixa ui;
24     string _atributo;
25
26     void inicilizarLabels();
27
28 private slots:
29     void emitirParametros();
30
31 signals:
32     void aplicarParamsDCTodos(const double, const double);
33
34 };
35
36 #endif // WIDGET_PARAM_DIAG_CAIXA_H
```

Listagem B.91: widget_param_diag_caixa.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>WidgetParamDiagCaixa</class>
4   <widget class="QWidget" name="WidgetParamDiagCaixa">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>403</width>
10     <height>168</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>WidgetParamDiagCaixa</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout">
17    <item>
18     <layout class="QFormLayout" name="formLayout">
19      <item row="0" column="0">
20       <widget class="QLabel" name="label">
21        <property name="text">
22         <string>Desvio interquartílico:</string>
23        </property>
24       </widget>
25      </item>
26      <item row="0" column="1">
27       <widget class="QLabel" name="labelDI">
28        <property name="text">
```

```
29          <string>TextLabel</string>
30         </property>
31        </widget>
32       </item>
33       <item row="1" column="0">
34        <widget class="QLabel" name="label_6">
35         <property name="text">
36          <string>Quartíl superior:</string>
37         </property>
38        </widget>
39       </item>
40       <item row="1" column="1">
41        <widget class="QLabel" name="labelQS">
42         <property name="text">
43          <string>TextLabel</string>
44         </property>
45        </widget>
46       </item>
47       <item row="2" column="0">
48        <widget class="QLabel" name="label_7">
49         <property name="text">
50          <string>Quartíl inferior:</string>
51         </property>
52        </widget>
53       </item>
54       <item row="2" column="1">
55        <widget class="QLabel" name="labelQI">
56         <property name="text">
57          <string>TextLabel</string>
58         </property>
59        </widget>
60       </item>
61      </layout>
62     </item>
63     <item>
64      <layout class="QHBoxLayout" name="horizontalLayout">
65       <item>
66        <widget class="QLabel" name="label_2">
67         <property name="text">
68          <string>Whisker superior = quartil superior + </string>
69         </property>
70         <property name="alignment">
71          <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
72         </property>
73        </widget>
74       </item>
75       <item>
76        <widget class="QDoubleSpinBox" name="doubleSpinBoxFatorWS">
77         <property name="maximum">
78          <double>999999999.000000000000000</double>
79         </property>
80         <property name="singleStep">
81          <double>0.100000000000000</double>
82         </property>
83         <property name="value">
84          <double>1.500000000000000</double>
85         </property>
86        </widget>
87       </item>
88       <item>
89        <widget class="QLabel" name="label_4">
90         <property name="text">
91          <string> * Desvio interquartílico</string>
92         </property>
93        </widget>
94       </item>
95       <item>
96        <spacer name="horizontalSpacer_3">
97         <property name="orientation">
98          <enum>Qt::Horizontal</enum>
99         </property>
100        <property name="sizeHint" stdset="0">
101         <size>
102          <width>40</width>
103          <height>20</height>
104         </size>
105        </property>
106       </spacer>
107      </item>
108     </layout>
109    </item>
110    <item>
```

```
111      <layout class="QHBoxLayout" name="horizontalLayout_2">
112       <item>
113        <spacer name="horizontalSpacer_4">
114         <property name="orientation">
115          <enum>Qt::Horizontal</enum>
116         </property>
117         <property name="sizeType">
118          <enum>QSizePolicy::Fixed</enum>
119         </property>
120         <property name="sizeHint" stdset="0">
121          <size>
122           <width>8</width>
123           <height>20</height>
124          </size>
125         </property>
126        </spacer>
127       </item>
128       <item>
129        <widget class="QLabel" name="label_3">
130         <property name="text">
131          <string>Whisker inferior = quartil inferior - </string>
132         </property>
133         <property name="alignment">
134          <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
135         </property>
136        </widget>
137       </item>
138       <item>
139        <widget class="QDoubleSpinBox" name="doubleSpinBoxFatorWI">
140         <property name="maximum">
141          <double>999999999.000000000000000</double>
142         </property>
143         <property name="singleStep">
144          <double>0.100000000000000</double>
145         </property>
146         <property name="value">
147          <double>1.500000000000000</double>
148         </property>
149        </widget>
150       </item>
151       <item>
152        <widget class="QLabel" name="label_5">
153         <property name="text">
154          <string> * Desvio interquartílico</string>
155         </property>
156        </widget>
157       </item>
158       <item>
159        <spacer name="horizontalSpacer_2">
160         <property name="orientation">
161          <enum>Qt::Horizontal</enum>
162         </property>
163         <property name="sizeHint" stdset="0">
164          <size>
165           <width>40</width>
166           <height>20</height>
167          </size>
168         </property>
169        </spacer>
170       </item>
171      </layout>
172     </item>
173     <item>
174      <layout class="QHBoxLayout" name="horizontalLayout_3">
175       <item>
176        <widget class="QPushButton" name="pushButtonAplicarTodos">
177         <property name="text">
178          <string>Aplicar estes parâmetros para os demais atributos</string>
179         </property>
180        </widget>
181       </item>
182       <item>
183        <spacer name="horizontalSpacer">
184         <property name="orientation">
185          <enum>Qt::Horizontal</enum>
186         </property>
187         <property name="sizeHint" stdset="0">
188          <size>
189           <width>40</width>
190           <height>20</height>
191          </size>
192         </property>
```

```
193         </spacer>
194        </item>
195       </layout>
196      </item>
197      <item>
198       <spacer name="verticalSpacer">
199        <property name="orientation">
200         <enum>Qt::Vertical</enum>
201        </property>
202        <property name="sizeHint" stdset="0">
203         <size>
204          <width>20</width>
205          <height>1</height>
206         </size>
207        </property>
208       </spacer>
209      </item>
210     </layout>
211    </widget>
212    <layoutdefault spacing="6" margin="11"/>
213    <resources/>
214    <connections/>
215   </ui>
```

Listagem B.92: widget_param_diag_caixa.ui

```
1  #include "widget_param_zscore.h"
2
3  #include "coletor_estatisticas.h"
4  #include "utilidades.h"
5
6  #include <QDebug>
7
8  WidgetParamZscore::WidgetParamZscore(const bool telaImportacao, const
       string &atributo, QWidget *parent)
9      : QWidget(parent), _atributo(atributo)
10 {
11     ui.setupUi(this);
12
13     inicializarLabels();
14
15     ui.doubleSpinBoxZmax->setValue(ColetorEstatisticas::_ZMAX);
16     ui.doubleSpinBoxZmin->setValue(ColetorEstatisticas::_ZMIN);
17
18     connect(ui.pushButtonAplicarTodos, SIGNAL(clicked()), this, SLOT(
           emitirParametros()));
19
20     if (!telaImportacao)
21     {
22         ui.pushButtonAplicarTodos->setVisible(false);
23         ui.horizontalSpacer->changeSize(0,0, QSizePolicy::Fixed,
               QSizePolicy::Fixed);
24     }
25 }
26
27 WidgetParamZscore::~WidgetParamZscore()
28 {
29     const void * address = static_cast<const void*>(&(*this));
30     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
31 }
32
33 void WidgetParamZscore::inicializarLabels()
34 {
35     const ColetorEstatisticas::EstatisticaQuant &estatistica =
           ColetorEstatisticas::obterInstancia()->
           estatisticaAtributoQuantitativo(_atributo);
36
37     ui.labelDP->setText(QString::fromLatin1(Utilidades::dblToStdStr(
           estatistica.desvioPadrao, 3).c_str()));
38     ui.labelMedia->setText(QString::fromLatin1(Utilidades::dblToStdStr(
           estatistica.media, 3).c_str()));
39 }
40
41
42 double WidgetParamZscore::zMin() const
43 {
44     return ui.doubleSpinBoxZmin->value();
45 }
46
```

```
47 double WidgetParamZscore::zMax() const
48 {
49     return ui.doubleSpinBoxZmax->value();
50 }
51
52 void WidgetParamZscore::setZMin(const double zMin)
53 {
54     ui.doubleSpinBoxZmin->setValue(zMin);
55 }
56
57 void WidgetParamZscore::setZMax(const double fatorWS)
58 {
59     ui.doubleSpinBoxZmax->setValue(fatorWS);
60 }
61
62 void WidgetParamZscore::emitirParametros()
63 {
64     emit aplicarParamsZSTodos(zMin(), zMax());
65 }
```

Listagem B.93: widget_param_zscore.cpp

```
1  #ifndef WIDGET_PARAM_ZSCORE_H
2  #define WIDGET_PARAM_ZSCORE_H
3
4  #include "ui_widget_param_zscore.h"
5
6  #include <string>
7
8  using namespace std;
9
10 class WidgetParamZscore : public QWidget
11 {
12     Q_OBJECT
13
14 public:
15     WidgetParamZscore(const bool telaImportacao, const string &atributo,
           QWidget *parent = 0);
16     ~WidgetParamZscore();
17     double zMin() const;
18     double zMax() const;
19     void setZMin(const double zMin);
20     void setZMax(const double zMax);
21
22
23 private:
24     Ui::WidgetParamZscore ui;
25     string _atributo;
26
27     void inicializarLabels();
28
29 private slots:
30     void emitirParametros();
31
32 signals:
33     void aplicarParamsZSTodos(const double, const double);
34 };
35
36 #endif // WIDGET_PARAM_ZSCORE_H
```

Listagem B.94: widget_param_zscore.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>WidgetParamZscore</class>
4   <widget class="QWidget" name="WidgetParamZscore">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>274</width>
10     <height>147</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>WidgetParamZscore</string>
```

```
15    </property>
16    <layout class="QVBoxLayout" name="verticalLayout_3">
17     <item>
18      <layout class="QHBoxLayout" name="horizontalLayout">
19       <item>
20        <layout class="QVBoxLayout" name="verticalLayout">
21         <item>
22          <widget class="QLabel" name="label">
23           <property name="text">
24            <string>Média:</string>
25           </property>
26          </widget>
27         </item>
28         <item>
29          <widget class="QLabel" name="label_2">
30           <property name="text">
31            <string>Desvio Padrão:</string>
32           </property>
33          </widget>
34         </item>
35         <item>
36          <widget class="QLabel" name="label_3">
37           <property name="minimumSize">
38            <size>
39             <width>0</width>
40             <height>20</height>
41            </size>
42           </property>
43           <property name="text">
44            <string>Z mínimo:</string>
45           </property>
46          </widget>
47         </item>
48         <item>
49          <widget class="QLabel" name="label_4">
50           <property name="minimumSize">
51            <size>
52             <width>0</width>
53             <height>20</height>
54            </size>
55           </property>
56           <property name="text">
57            <string>Z máximo:</string>
58           </property>
59          </widget>
60         </item>
61        </layout>
62       </item>
63       <item>
64        <layout class="QVBoxLayout" name="verticalLayout_2">
65         <item>
66          <widget class="QLabel" name="labelMedia">
67           <property name="text">
68            <string>TextLabel</string>
69           </property>
70          </widget>
71         </item>
72         <item>
73          <widget class="QLabel" name="labelDP">
74           <property name="text">
75            <string>TextLabel</string>
76           </property>
77          </widget>
78         </item>
79         <item>
80          <widget class="QDoubleSpinBox" name="doubleSpinBoxZmin">
81           <property name="minimum">
82            <double>-999999999.000000000000000</double>
83           </property>
84           <property name="maximum">
85            <double>0.000000000000000</double>
86           </property>
87           <property name="value">
88            <double>-2.000000000000000</double>
89           </property>
90          </widget>
91         </item>
92         <item>
93          <widget class="QDoubleSpinBox" name="doubleSpinBoxZmax">
94           <property name="maximum">
95            <double>999999999.000000000000000</double>
96           </property>
```

```
97            <property name="value">
98             <double>2.000000000000000</double>
99            </property>
100           </widget>
101          </item>
102         </layout>
103        </item>
104        <item>
105         <spacer name="horizontalSpacer_2">
106          <property name="orientation">
107           <enum>Qt::Horizontal</enum>
108          </property>
109          <property name="sizeHint" stdset="0">
110           <size>
111            <width>40</width>
112            <height>20</height>
113           </size>
114          </property>
115         </spacer>
116        </item>
117       </layout>
118      </item>
119      <item>
120       <layout class="QHBoxLayout" name="horizontalLayout_3">
121        <item>
122         <widget class="QPushButton" name="pushButtonAplicarTodos">
123          <property name="text">
124           <string>Aplicar estes parâmetros para os demais atributos</string>
125          </property>
126         </widget>
127        </item>
128        <item>
129         <spacer name="horizontalSpacer">
130          <property name="orientation">
131           <enum>Qt::Horizontal</enum>
132          </property>
133          <property name="sizeHint" stdset="0">
134           <size>
135            <width>40</width>
136            <height>20</height>
137           </size>
138          </property>
139         </spacer>
140        </item>
141       </layout>
142      </item>
143      <item>
144       <spacer name="verticalSpacer">
145        <property name="orientation">
146         <enum>Qt::Vertical</enum>
147        </property>
148        <property name="sizeHint" stdset="0">
149         <size>
150          <width>20</width>
151          <height>40</height>
152         </size>
153        </property>
154       </spacer>
155      </item>
156     </layout>
157    </widget>
158    <layoutdefault spacing="6" margin="11"/>
159    <resources/>
160    <connections/>
161   </ui>
```

Listagem B.95: widget_param_zscore.ui

```
1  #include "widget_parametro.h"
2
3  #include <QDebug>
4
5  WidgetParametro::WidgetParametro(
6      const string &nomeParametro,
7      const double valorInicial,
8      QWidget *parent)
9      : QWidget(parent), _valorInicial(valorInicial)
10 {
11     ui.setupUi(this);
```

```
12
13      ui.labelParam->setText(QString::fromLatin1(nomeParametro.c_str()));
14
15      double step = abs(valorInicial) * 0.05; //10%
16      if (step < 0.01) {step = 0.01;}
17
18      ui.doubleSpinBoxParametro->setSingleStep(step);
19      ui.doubleSpinBoxParametro->setValue(_valorInicial);
20
21      connect(ui.pushButtonValorInicial, SIGNAL(clicked()), this, SLOT(
            valorInicialClick()));
22  }
23
24  WidgetParametro::~WidgetParametro()
25  {
26      const void * address = static_cast<const void*>(&(*this));
27      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
28  }
29
30  double WidgetParametro::valorParametro() const
31  {
32      return ui.doubleSpinBoxParametro->value();
33  }
34
35  void WidgetParametro::setValorParametro(const double valor)
36  {
37      ui.doubleSpinBoxParametro->setValue(valor);
38  }
39
40  void WidgetParametro::valorInicialClick()
41  {
42      ui.doubleSpinBoxParametro->setValue(_valorInicial);
43  }
```

Listagem B.96: widget_parametro.cpp

```
1  #ifndef WIDGET_PARAMETRO_H
2  #define WIDGET_PARAMETRO_H
3
4  #include "ui_widget_parametro.h"
5
6  #include <string>
7
8  using namespace std;
9
10 class WidgetParametro : public QWidget
11 {
12     Q_OBJECT
13
14 public:
15     WidgetParametro(
16         const string &nomeParametro,
17         const double valorInicial,
18         QWidget *parent = 0);
19     ~WidgetParametro();
20
21     double valorParametro() const;
22     void setValorParametro(const double valor);
23
24 private:
25     Ui::WidgetParametro ui;
26     double _valorInicial;
27
28 private slots:
29     void valorInicialClick();
30
31 };
32
33 #endif // WIDGET_PARAMETRO_H
```

Listagem B.97: widget_parametro.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>WidgetParametro</class>
4   <widget class="QWidget" name="WidgetParametro">
```

```
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>241</width>
10     <height>23</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>WidgetParametro</string>
15   </property>
16   <layout class="QHBoxLayout" name="horizontalLayout">
17    <property name="leftMargin">
18     <number>0</number>
19    </property>
20    <property name="topMargin">
21     <number>0</number>
22    </property>
23    <property name="rightMargin">
24     <number>0</number>
25    </property>
26    <property name="bottomMargin">
27     <number>0</number>
28    </property>
29    <item>
30     <widget class="QLabel" name="labelParam">
31      <property name="minimumSize">
32       <size>
33        <width>75</width>
34        <height>0</height>
35       </size>
36      </property>
37      <property name="text">
38       <string>TextLabel</string>
39      </property>
40      <property name="alignment">
41       <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
42      </property>
43     </widget>
44    </item>
45    <item>
46     <widget class="QDoubleSpinBox" name="doubleSpinBoxParametro">
47      <property name="minimum">
48       <double>-999999999.000000000000000</double>
49      </property>
50      <property name="maximum">
51       <double>999999999.000000000000000</double>
52      </property>
53     </widget>
54    </item>
55    <item>
56     <widget class="QPushButton" name="pushButtonValorInicial">
57      <property name="text">
58       <string>Valor inicial</string>
59      </property>
60     </widget>
61    </item>
62   </layout>
63  </widget>
64  <layoutdefault spacing="6" margin="11"/>
65  <resources/>
66  <connections/>
67 </ui>
```

Listagem B.98: widget_parametro.ui

```
1  #include "widget_regressao_linear_simples.h"
2
3  #include "tabela_registros.h"
4  #include "populador.h"
5
6  #include <QDebug>
7
8  WidgetRegressaoLinearSimples::WidgetRegressaoLinearSimples(
9      const string &atributoDependente,
10     const vector<string> &todosAtributos,
11     const vector<vector<size_t>> &indicesValidosPorAtributo,
12     QWidget *parent)
```

```
13      : QWidget(parent), _atributoDependente(atributoDependente),
            _atributosInd(todosAtributos), _indicesValidosPorAtributoInd(
            indicesValidosPorAtributo)
14  {
15      ui.setupUi(this);
16
17      ui.labelDependente->setText(QString::fromLatin1(atributoDependente.
            c_str()));
18
19      vector<string>::const_iterator citAtributoDep = std::find(_atributosInd
            .begin(), _atributosInd.end(), _atributoDependente);
20      const size_t atributoDepIndice = citAtributoDep - _atributosInd.cbegin
            ();
21      _indicesValidosAtributoDep = indicesValidosPorAtributo[
            atributoDepIndice];
22      _atributosInd.erase(citAtributoDep);
23      _indicesValidosPorAtributoInd.erase(_indicesValidosPorAtributoInd.begin
            () + atributoDepIndice);
24
25      Populador::populeComboBox(ui.comboBoxAtributoIndep, _atributosInd);
26
27      connect(ui.comboBoxAtributoIndep, SIGNAL(currentIndexChanged(const
            QString&)), this, SLOT(currentIndexChangedComboBoxIndependente()))
            ;
28
29      currentIndexChangedComboBoxIndependente();
30  }
31
32  WidgetRegressaoLinearSimples::~WidgetRegressaoLinearSimples()
33  {
34      const void * address = static_cast<const void*>(&(*this));
35      qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
36  }
37
38  const RegressaoLinearSimples &WidgetRegressaoLinearSimples::
        regressaoLinearSimples() const
39  {
40      return _regressaoLinearSimples;
41  }
42
43  string WidgetRegressaoLinearSimples::atributoIndep() const
44  {
45      return ui.comboBoxAtributoIndep->currentText().toLatin1().data();
46  }
47
48  void WidgetRegressaoLinearSimples::currentIndexChangedComboBoxIndependente
        ()
49  {
50      const string atributoIndependente = ui.comboBoxAtributoIndep->
            currentText().toLatin1().data();
51
52      vector<string> atributos = vector<string>();
53      atributos.push_back(atributoIndependente);
54      atributos.push_back(_atributoDependente);
55
56      std::sort(_indicesValidosAtributoDep.begin(),
            _indicesValidosAtributoDep.begin());
57
58      vector<string>::const_iterator citAtributoInd = std::find(_atributosInd
            .begin(), _atributosInd.end(), atributoIndependente);
59      const size_t indiceAtrbInd = citAtributoInd - _atributosInd.cbegin();
60      vector<size_t> &indicesValidosAtributoInd =
            _indicesValidosPorAtributoInd[indiceAtrbInd];
61      std::sort(indicesValidosAtributoInd.begin(), indicesValidosAtributoInd.
            begin());
62
63      vector<size_t> indicesInterseccao = vector<size_t>();
64      set_intersection(_indicesValidosAtributoDep.begin(),
65          _indicesValidosAtributoDep.end(),
66          indicesValidosAtributoInd.begin(),
67          indicesValidosAtributoInd.end(),
68          inserter(indicesInterseccao, indicesInterseccao.begin()));
69
70      vector<vector<double>> valoresColunas = vector<vector<double>>();
71      TabelaDeRegistros::obterInstancia()->valoresAtributosQuantDblPorIndice(
            atributos, indicesInterseccao, false, valoresColunas);
72
73      _regressaoLinearSimples = RegressaoLinearSimples(valoresColunas[0],
            valoresColunas[1]);
74
75      ui.doubleSpinBoxAlpha->setValue(_regressaoLinearSimples.alpha());
76      ui.doubleSpinBoxBeta->setValue(_regressaoLinearSimples.beta());
```

```
77 }
```

Listagem B.99: widget_regressao_linear_simples.cpp

```
1  #ifndef WIDGET_REGRESSAO_LINEAR_SIMPLES_H
2  #define WIDGET_REGRESSAO_LINEAR_SIMPLES_H
3
4  #include <QWidget>
5  #include "ui_widget_regressao_linear_simples.h"
6
7  #include "regressao_linear_simples.h"
8
9  #include <string>
10
11 using namespace std;
12
13 class WidgetRegressaoLinearSimples : public QWidget
14 {
15     Q_OBJECT
16
17 public:
18     WidgetRegressaoLinearSimples(
19         const string &atributoDependente,
20         const vector<string> &todosAtributos,
21         const vector<vector<size_t>> &indicesValidosPorAtributo,
22         QWidget *parent = 0);
23     ~WidgetRegressaoLinearSimples();
24
25     const RegressaoLinearSimples &regressaoLinearSimples() const;
26     string atributoIndep() const;
27
28 private:
29     Ui::WidgetRegressaoLinearSimples ui;
30     string _atributoDependente;
31     vector<size_t> _indicesValidosAtributoDep;
32
33     vector<string> _atributosInd;
34     vector<vector<size_t>> _indicesValidosPorAtributoInd;
35
36     RegressaoLinearSimples _regressaoLinearSimples;
37
38 private slots:
39     void currentIndexChangedComboBoxIndependente();
40 };
41
42 #endif // WIDGET_REGRESSAO_LINEAR_SIMPLES_H
```

Listagem B.100: widget_regressao_linear_simples.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>WidgetRegressaoLinearSimples</class>
4   <widget class="QWidget" name="WidgetRegressaoLinearSimples">
5    <property name="geometry">
6     <rect>
7      <x>0</x>
8      <y>0</y>
9      <width>400</width>
10     <height>110</height>
11    </rect>
12   </property>
13   <property name="windowTitle">
14    <string>WidgetRegressaoLinearSimples</string>
15   </property>
16   <layout class="QVBoxLayout" name="verticalLayout_2">
17    <property name="spacing">
18     <number>0</number>
19    </property>
20    <property name="leftMargin">
21     <number>0</number>
22    </property>
23    <property name="topMargin">
24     <number>0</number>
25    </property>
26    <property name="rightMargin">
27     <number>0</number>
```

```
28      </property>
29      <property name="bottomMargin">
30       <number>0</number>
31      </property>
32      <item>
33       <widget class="QGroupBox" name="groupBox">
34        <property name="title">
35         <string>Regressão linear</string>
36        </property>
37        <layout class="QVBoxLayout" name="verticalLayout">
38         <item>
39          <layout class="QFormLayout" name="formLayout">
40           <item row="0" column="0">
41            <widget class="QLabel" name="label">
42             <property name="sizePolicy">
43              <sizepolicy hsizetype="Fixed" vsizetype="Preferred">
44               <horstretch>0</horstretch>
45               <verstretch>0</verstretch>
46              </sizepolicy>
47             </property>
48             <property name="text">
49              <string>Conjunto dependente (Y):</string>
50             </property>
51            </widget>
52           </item>
53           <item row="0" column="1">
54            <widget class="QLabel" name="labelDependente">
55             <property name="text">
56              <string>TextLabel</string>
57             </property>
58            </widget>
59           </item>
60           <item row="1" column="0">
61            <widget class="QLabel" name="label_2">
62             <property name="sizePolicy">
63              <sizepolicy hsizetype="Fixed" vsizetype="Preferred">
64               <horstretch>0</horstretch>
65               <verstretch>0</verstretch>
66              </sizepolicy>
67             </property>
68             <property name="text">
69              <string>Conjunto independente (X):</string>
70             </property>
71            </widget>
72           </item>
73           <item row="1" column="1">
74            <widget class="QComboBox" name="comboBoxAtributoIndep">
75             <property name="sizePolicy">
76              <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
77               <horstretch>0</horstretch>
78               <verstretch>0</verstretch>
79              </sizepolicy>
80             </property>
81             <property name="minimumSize">
82              <size>
83               <width>150</width>
84               <height>0</height>
85              </size>
86             </property>
87            </widget>
88           </item>
89          </layout>
90         </item>
91         <item>
92          <layout class="QHBoxLayout" name="horizontalLayout">
93           <item>
94            <widget class="QLabel" name="label_3">
95             <property name="text">
96              <string>Yi = </string>
97             </property>
98             <property name="alignment">
99              <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
100            </property>
101           </widget>
102          </item>
103          <item>
104           <widget class="QDoubleSpinBox" name="doubleSpinBoxAlpha">
105            <property name="minimumSize">
106             <size>
107              <width>75</width>
108              <height>0</height>
109             </size>
```

```
110             </property>
111             <property name="decimals">
112              <number>10</number>
113             </property>
114             <property name="minimum">
115              <double>-999999999.000000000000000</double>
116             </property>
117             <property name="maximum">
118              <double>999999999.000000000000000</double>
119             </property>
120            </widget>
121           </item>
122           <item>
123            <widget class="QLabel" name="label_4">
124             <property name="text">
125              <string> + </string>
126             </property>
127             <property name="alignment">
128              <set>Qt::AlignCenter</set>
129             </property>
130            </widget>
131           </item>
132           <item>
133            <widget class="QDoubleSpinBox" name="doubleSpinBoxBeta">
134             <property name="minimumSize">
135              <size>
136               <width>75</width>
137               <height>0</height>
138              </size>
139             </property>
140             <property name="decimals">
141              <number>10</number>
142             </property>
143             <property name="minimum">
144              <double>-999999999.000000000000000</double>
145             </property>
146             <property name="maximum">
147              <double>999999999.000000000000000</double>
148             </property>
149            </widget>
150           </item>
151           <item>
152            <widget class="QLabel" name="label_5">
153             <property name="text">
154              <string>* Xi</string>
155             </property>
156             <property name="alignment">
157              <set>Qt::AlignCenter</set>
158             </property>
159            </widget>
160           </item>
161           <item>
162            <spacer name="horizontalSpacer_2">
163             <property name="orientation">
164              <enum>Qt::Horizontal</enum>
165             </property>
166             <property name="sizeHint" stdset="0">
167              <size>
168               <width>40</width>
169               <height>20</height>
170              </size>
171             </property>
172            </spacer>
173           </item>
174          </layout>
175         </item>
176         <item>
177          <spacer name="verticalSpacer">
178           <property name="orientation">
179            <enum>Qt::Vertical</enum>
180           </property>
181           <property name="sizeHint" stdset="0">
182            <size>
183             <width>20</width>
184             <height>40</height>
185            </size>
186           </property>
187          </spacer>
188         </item>
189        </layout>
190       </widget>
191      </item>
```

```
192   </layout>
193  </widget>
194  <layoutdefault spacing="6" margin="11"/>
195  <resources/>
196  <connections/>
197 </ui>
```

Listagem B.101: widget_regressao_linear_simples.ui

```
1  #include "widget_tabela_editora.h"
2
3  #include <QAbstractItemView>
4  #include <QAbstractButton>
5  #include <QDebug>
6  #include <QMenu>
7  #include <QMenuBar>
8
9  #include "dialog_subst_qual.h"
10 #include "dialog_subst_quant.h"
11 #include "dialog_renomear.h"
12 #include "dialog_nova_coluna.h"
13 #include "dialog_escolher_distribuicao.h"
14 #include "dialog_escolher_RLS.h"
15 #include "dialog_rem_ide_quali.h"
16 #include "dialog_rem_ide_quant.h"
17 #include "dialog_rem_ide_outliers.h"
18
19 #include "coletor_estatisticas.h"
20 #include "populador.h"
21 #include "utilidades.h"
22 #include "thread_worker_coletar_estatisticas.h"
23
24 WidgetTabelaEditora::WidgetTabelaEditora(const vector<size_t> &
       indicesRegistros, QWidget *parent)
25     : WidgetProgresso(parent), _comboBox(NULL), _indiceCombo(NULL),
           _totalLinhas(indicesRegistros.size()),
26     _tableViewEdicaoModel(new QStandardItemModel(this)), _linhaClick(-1),
           _colunaClick(-1),
27     _indicesTodosRegistros(indicesRegistros),
           _registrosDestacadosPorAtributo()
28 {
29     ui.setupUi(this);
30
31     ui.tableViewEdicao->setModel(_tableViewEdicaoModel);
32     ui.tableViewEdicao->setSelectionMode(QAbstractItemView::SingleSelection
           );
33     ui.tableViewEdicao->setSortingEnabled(true);
34     ui.tableViewEdicao->verticalHeader()->setVisible(false);
35
36     vector<vector<string>> registrosVetor = vector<vector<string>>();
37     TabelaDeRegistros::obterInstancia()->registrosVetorStrPorIndice(
           _indicesTodosRegistros, registrosVetor, true);
38
39     vector<string> atributosCopia = TabelaDeRegistros::obterInstancia()->
           atributos();
40     atributosCopia.insert(atributosCopia.begin(), "Índice do registro");
41
42     Populador::populeTabela(ui.tableViewEdicao, atributosCopia,
           registrosVetor);
43     ui.tableViewEdicao->setColumnHidden(0, true); //esconde a primeira
           coluna que representa o Índice do registro
44
45     inicializaMenuBarra();
46
47     ui.progressBar->setValue(0);
48
49     connect(ui.tableViewEdicao, SIGNAL(clicked(const QModelIndex&)), this,
           SLOT(celulaClick(const QModelIndex&)));
50     connect(ui.tableViewEdicao, SIGNAL(doubleClicked(const QModelIndex&)),
           this, SLOT(celulaClickDuplo(const QModelIndex&)));
51     connect(_tableViewEdicaoModel, SIGNAL(itemChanged(QStandardItem*)),
           this, SLOT(celulaAlterada(QStandardItem*)));
52 }
53
54 WidgetTabelaEditora::~WidgetTabelaEditora()
55 {
56     const void * address = static_cast<const void*>(&(*this));
57     qDebug() << __FILE__ << " " << __LINE__ << " Destrutor " << address;
58
```

```
59      if (_comboBox) delete _comboBox;
60      if (_indiceCombo) delete _indiceCombo;
61      if (_tableViewEdicaoModel) delete _tableViewEdicaoModel;
62  }
63
64  void WidgetTabelaEditora::inicializaMenuBarra()
65  {
66      QAction *removerLinhaAct;
67      QAction *removerColunaAct;
68      QAction *inserirLinhaAct;
69      QAction *inserirColunaAct;
70      QAction *mostrarTodosAct;
71      QAction *remLocValorAct;
72      QAction *substituaAct;
73      QAction *renomearAct;
74      QAction *gerarValorDP;
75      QAction *popularLinhaDP;
76      QAction *popularColunaDP;
77      QAction *remIdeOutliers;
78      QAction *gerarValorRLS;
79      QAction *popularLinhaRLS;
80      QAction *popularColunaRLS;
81
82      //menu editar
83      QMenu *menuEditar = new QMenu("Editar", this);
84      remLocValorAct = new QAction("Remover/Localizar Registros", this);
85      substituaAct = new QAction("Substituir Valores", this);
86      renomearAct = new QAction("Renomear Atributo", this);
87      connect(remLocValorAct, SIGNAL(triggered()), this, SLOT(
            removerIdentificarClick()));
88      connect(substituaAct, SIGNAL(triggered()), this, SLOT(substituaClick())
            );
89      connect(renomearAct, SIGNAL(triggered()), this, SLOT(
            renomearColunaDialog()));
90      menuEditar->addAction(remLocValorAct);
91      menuEditar->addAction(substituaAct);
92      menuEditar->addAction(renomearAct);
93  #ifndef TCC_RODOLFO
94      renomearAct->setEnabled(false); //Não se pode renomear atributos dos
            dados do poço. Garantir consistência
95  #endif
96
97      //menu remover
98      QMenu *menuRemover = new QMenu("Remover", this);
99      removerLinhaAct = new QAction("Linha", this);
100     removerColunaAct = new QAction("Coluna", this);
101     connect(removerLinhaAct, SIGNAL(triggered()), this, SLOT(removerLinha()
            ));
102     connect(removerColunaAct, SIGNAL(triggered()), this, SLOT(removerColuna
            ()));
103     menuRemover->addAction(removerLinhaAct);
104     menuRemover->addAction(removerColunaAct);
105 #ifndef TCC_RODOLFO
106     removerColunaAct->setEnabled(false); //Não se pode remover atributos
            dos dados do poço. Garantir consistência
107 #endif
108
109     //menu inserir
110     QMenu *menuInserir = new QMenu("Inserir", this);
111     inserirLinhaAct = new QAction("Linha", this);
112     inserirColunaAct = new QAction("Coluna", this);
113     connect(inserirLinhaAct, SIGNAL(triggered()), this, SLOT(inserirLinha()
            ));
114     connect(inserirColunaAct, SIGNAL(triggered()), this, SLOT(
            inserirColunaDialog()));
115     menuInserir->addAction(inserirLinhaAct);
116     menuInserir->addAction(inserirColunaAct);
117 #ifndef TCC_RODOLFO
118     inserirColunaAct->setEnabled(false); //Não se pode inserir atributos
            nos dados do poço. Garantir consistência
119 #endif
120
121     //ferramentas
122     QMenu *menuFerramentas = new QMenu("Ferramentas", this);
123     gerarValorDP = new QAction("Gerar Valor de uma DP", this);
124     popularLinhaDP = new QAction("Popular linha com DPs", this);
125     popularColunaDP = new QAction("Popular coluna com uma DP", this);
126     remIdeOutliers = new QAction("Remover/Identificar outliers", this);
127     gerarValorRLS = new QAction("Gerar Valor de uma RLS", this);
128     popularLinhaRLS = new QAction("Popular linha com RLSs", this);
129     popularColunaRLS = new QAction("Popular coluna com uma RLS", this);
130     connect(gerarValorDP, SIGNAL(triggered()), this, SLOT(
```

```
            gerarValorDPDialog()));
131     connect(popularLinhaDP, SIGNAL(triggered()), this, SLOT(
            gerarLinhaDPDialog()));
132     connect(popularColunaDP, SIGNAL(triggered()), this, SLOT(
            gerarColunaDPDialog()));
133     connect(remIdeOutliers, SIGNAL(triggered()), this, SLOT(
            remIdeOutliersDialog()));
134     connect(gerarValorRLS, SIGNAL(triggered()), this, SLOT(
            gerarValorRLSDialog()));
135     connect(popularLinhaRLS, SIGNAL(triggered()), this, SLOT(
            gerarLinhaRLSDialog()));
136     connect(popularColunaRLS, SIGNAL(triggered()), this, SLOT(
            gerarColunaRLSDialog()));
137     menuFerramentas->addAction(gerarValorDP);
138     menuFerramentas->addAction(popularLinhaDP);
139     menuFerramentas->addAction(popularColunaDP);
140     menuFerramentas->addAction(remIdeOutliers);
141     menuFerramentas->addAction(gerarValorRLS);
142     menuFerramentas->addAction(popularLinhaRLS);
143     menuFerramentas->addAction(popularColunaRLS);
144
145     mostrarTodosAct = new QAction("Mostrar Todos os Registros", this);
146     connect(mostrarTodosAct, SIGNAL(triggered()), this, SLOT(mostrarTodos()
            ));
147
148     //menu principal
149     _menu = new QMenuBar(this);
150     _menu->resize(this->width(), _menu->height());
151     //_menu->addMenu(menuExportar);
152     _menu->addMenu(menuEditar);
153     _menu->addMenu(menuRemover);
154     _menu->addMenu(menuInserir);
155     _menu->addMenu(menuFerramentas);
156     _menu->addAction(mostrarTodosAct);
157 }
158
159 void WidgetTabelaEditora::habilitarEventos(const bool habilitado) const
160 {
161     _menu->setEnabled(habilitado);
162     ui.tableViewEdicao->setEnabled(habilitado);
163 }
164
165 void WidgetTabelaEditora::removerIdentificarClick()
166 {
167     removerComboBox();
168
169     const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
170     if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
            contem os ids
171     {
172         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Selecione uma coluna!"));
173         return;
174     }
175     const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
            Qt::Horizontal).toString().toLatin1().data();
176
177     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributo))
178     {
179         removerIdentificarQualitativaDialog(atributo);
180     }
181     else
182     {
183         removerIdentificarQuantitativoDialog(atributo);
184     }
185 }
186
187 void WidgetTabelaEditora::removerIdentificarQualitativaDialog(const string
        &atributoSelecionado)
188 {
189     DialogRemIdeQuali *dialogRemIdeQuali = new DialogRemIdeQuali(
            atributoSelecionado, this);
190     connect(dialogRemIdeQuali, SIGNAL(removerRegistrosPorValor(const vector
            <ValorBase *const>&, const string&, const bool, const bool)), this
            , SLOT(removerRegistrosPorValor(const vector<ValorBase *const>&,
            const string&, const bool, const bool)));
191     connect(dialogRemIdeQuali, SIGNAL(identificarRegistrosPorValor(const
            vector<ValorBase *const>&, const string&, const bool, const bool))
            , this, SLOT(identificarRegistrosPorValor(const vector<ValorBase *
            const>&, const string&, const bool, const bool)));
192
193     dialogRemIdeQuali->exec();
```

```
194     delete dialogRemIdeQuali;
195 }
196
197 void WidgetTabelaEditora::removerIdentificarQuantitativoDialog(const string
        &atributoSelecionado)
198 {
199     const ColetorEstatisticas::EstatisticaQuant &estatisticaQuantitativo =
            ColetorEstatisticas::obterInstancia()->
            estatisticaAtributoQuantitativo(atributoSelecionado);
200
201     DialogRemIdeQuant *dialogRemIdeQuant = new DialogRemIdeQuant(
            atributoSelecionado, estatisticaQuantitativo.maximo,
            estatisticaQuantitativo.minimo, this);
202     connect(dialogRemIdeQuant, SIGNAL(removerRegistrosPorValor(const vector
            <ValorBase *const>&, const string&, const bool, const bool)), this
            , SLOT(removerRegistrosPorValor(const vector<ValorBase *const>&,
            const string&, const bool, const bool)));
203     connect(dialogRemIdeQuant, SIGNAL(removerRegistrosQuantPorIntervalo(
            const vector<TabelaDeRegistros::Intervalo>&, const string&, const
            bool, const bool)), this, SLOT(removerRegistrosQuantPorIntervalo(
            const vector<TabelaDeRegistros::Intervalo>&, const string&, const
            bool, const bool)));
204     connect(dialogRemIdeQuant, SIGNAL(identificarRegistrosPorValor(const
            vector<ValorBase *const>&, const string&, const bool, const bool))
            , this, SLOT(identificarRegistrosPorValor(const vector<ValorBase *
            const>&, const string&, const bool, const bool)));
205     connect(dialogRemIdeQuant, SIGNAL(identificarRegistrosQuantPorIntervalo
            (const vector<TabelaDeRegistros::Intervalo>&, const string&, const
             bool, const bool)), this, SLOT(
            identificarRegistrosQuantPorIntervalo(const vector<
            TabelaDeRegistros::Intervalo>&, const string&, const bool, const
            bool)));
206
207     dialogRemIdeQuant->exec();
208     delete dialogRemIdeQuant;
209 }
210
211 void WidgetTabelaEditora::remIdeOutliersDialog()
212 {
213     const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
214     if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
            contem os ids
215     {
216         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Selecione uma coluna!"));
217         return;
218     }
219     const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
             Qt::Horizontal).toString().toLatin1().data();
220
221     DialogRemIdeOutliers *dialogRemIdeOutliers = new DialogRemIdeOutliers(
            atributo, this);
222     connect(dialogRemIdeOutliers, SIGNAL(removerRegistrosPorValor(const
            vector<ValorBase *const>&, const string&, const bool, const bool))
            , this, SLOT(removerRegistrosPorValor(const vector<ValorBase *
            const>&, const string&, const bool, const bool)));
223     connect(dialogRemIdeOutliers, SIGNAL(identificarRegistrosPorValor(const
             vector<ValorBase *const>&, const string&, const bool, const bool)
            ), this, SLOT(identificarRegistrosPorValor(const vector<ValorBase
            *const>&, const string&, const bool, const bool)));
224
225     dialogRemIdeOutliers->exec();
226     delete dialogRemIdeOutliers;
227 }
228
229 void WidgetTabelaEditora::removerRegistrosPorValor(
230         const vector<ValorBase *const> &valores,
231         const string &atributo,
232         const bool missingsTmb,
233         const bool invalidosTmb)
234 {
235     vector<size_t> registrosRemovidos = vector<size_t>();
236
237     TabelaDeRegistros::obterInstancia()->removaRegistrosPorValores(atributo
            , valores, missingsTmb, invalidosTmb, registrosRemovidos);
238
239     aposRemoverRegistro(registrosRemovidos);
240 }
241
242 void WidgetTabelaEditora::removerRegistrosQuantPorIntervalo(
243     const vector<TabelaDeRegistros::Intervalo> &intervalos,
244     const string &atributo,
```

```
245     const bool missingsTmb,
246     const bool invalidosTmb)
247 {
248     vector<size_t> registrosRemovidos = vector<size_t>();
249
250     TabelaDeRegistros::obterInstancia()->removerRegistrosQuantPorIntervalo(
            intervalos, atributo, missingsTmb, invalidosTmb,
            registrosRemovidos);
251
252     aposRemoverRegistro(registrosRemovidos);
253 }
254
255 void WidgetTabelaEditora::identificarRegistrosPorValor(
256     const vector<ValorBase *const> &valores,
257     const string &atributo,
258     const bool missingsTmb,
259     const bool invalidosTmb) const
260 {
261     vector<size_t> registrosIdentificados = vector<size_t>();
262
263     TabelaDeRegistros::obterInstancia()->identificarRegistrosPorValor(
            atributo, valores, missingsTmb, invalidosTmb,
            registrosIdentificados);
264
265     aposIdentificarRegistro(registrosIdentificados);
266 }
267
268 void WidgetTabelaEditora::identificarRegistrosQuantPorIntervalo(
269     const vector<TabelaDeRegistros::Intervalo> &intervalos,
270     const string &atributo,
271     const bool missingsTmb,
272     const bool invalidosTmb) const
273 {
274     vector<size_t> registrosIdentificados = vector<size_t>();
275
276     TabelaDeRegistros::obterInstancia()->
            identificarRegistrosQuantPorIntervalo(atributo, intervalos,
            missingsTmb, invalidosTmb, registrosIdentificados);
277
278     aposIdentificarRegistro(registrosIdentificados);
279 }
280
281 void WidgetTabelaEditora::substituaClick()
282 {
283     removerComboBox();
284
285     const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
286     if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
            contem os ids
287     {
288         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Selecione uma coluna!"));
289         return;
290     }
291     const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
            Qt::Horizontal).toString().toLatin1().data();
292
293     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributo))
294     {
295         substituaQualitativoDialog(atributo);
296     }
297     else
298     {
299         substituaQuantitativoDialog(atributo);
300     }
301 }
302
303 void WidgetTabelaEditora::substituaQualitativoDialog(const string &
        atributoSelecionado)
304 {
305     vector<string> categoriasAtributoQualitativo = vector<string>();
306     ColetorEstatisticas::obterInstancia()->todasCategoriasAtributoQual(
            atributoSelecionado, categoriasAtributoQualitativo);
307
308     const vector<string> &atributos = TabelaDeRegistros::obterInstancia()->
            atributos();
309
310     bool algumAtributoComValorDestacado = false;
311
312     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
313         citAtributo != atributos.cend();
314         citAtributo++)
```

```
315     {
316         const string &atributo = *citAtributo;
317         if (_registrosDestacadosPorAtributo[atributo].size() > 0)
318         {
319             algumAtributoComValorDestacado = true;
320         }
321     }
322
323     DialogSubstQual *dialogSubstQual = new DialogSubstQual(
            categoriasAtributoQualitativo, atributoSelecionado, this); //
            deletado abaixo
324     dialogSubstQual->somenteDestacados(algumAtributoComValorDestacado);
325
326
327     connect(dialogSubstQual, SIGNAL(substituaMissings(const string&,
            ValorBase *const)), this, SLOT(substituaMissings(const string&,
            ValorBase *const)));
328     connect(dialogSubstQual, SIGNAL(substituaTodos(const string&, ValorBase
             *const)), this, SLOT(substituaTodos(const string&, ValorBase *
            const)));
329     connect(dialogSubstQual, SIGNAL(substituaDestacados(const string&,
            ValorBase *const)), this, SLOT(substituaDestacados(const string&,
            ValorBase *const)));
330     connect(dialogSubstQual, SIGNAL(substituaLocalizados(const string&,
            const vector<ValorBase *const>&, ValorBase *const, const bool,
            const bool)), this, SLOT(substituaLocalizados(const string&, const
             vector<ValorBase *const>&, ValorBase *const, const bool, const
            bool)));
331
332     dialogSubstQual->exec();
333     delete dialogSubstQual;
334 }
335
336 void WidgetTabelaEditora::substituaQuantitativoDialog(const string &
        atributoSelecionado)
337 {
338     const vector<string> &atributos = TabelaDeRegistros::obterInstancia()->
            atributos();
339     const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
            ()->atributosQuantitativos();
340
341     vector<vector<size_t>> registrosDestacadosPorAtributoVetor = vector<
            vector<size_t>>();
342     vector<vector<size_t>> indicesValidosPorAtributo = vector<vector<size_t
            >>();
343
344     bool algumAtributoComValorDestacado = false;
345
346     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
347         citAtributo != atributos.cend();
348         citAtributo++)
349     {
350         const string &atributo = *citAtributo;
351         registrosDestacadosPorAtributoVetor.push_back(
                _registrosDestacadosPorAtributo[atributo]);
352
353         if (_registrosDestacadosPorAtributo[atributo].size() > 0)
354         {
355             algumAtributoComValorDestacado = true;
356         }
357     }
358
359     TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
            registrosDestacadosPorAtributoVetor, indicesValidosPorAtributo);
360
361     DialogSubstQuant *dialogSubstQuant = new DialogSubstQuant(
            atributoSelecionado, atributosQuant, indicesValidosPorAtributo,
            this);
362     dialogSubstQuant->somenteDestacados(algumAtributoComValorDestacado);
363
364     connect(dialogSubstQuant, SIGNAL(substituaMissings(const string&,
            ValorBase *const)), this, SLOT(substituaMissings(const string&,
            ValorBase *const)));
365     connect(dialogSubstQuant, SIGNAL(substituaTodos(const string&,
            ValorBase *const)), this, SLOT(substituaTodos(const string&,
            ValorBase *const)));
366     connect(dialogSubstQuant, SIGNAL(substituaInvalidos(const string&,
            ValorBase *const)), this, SLOT(substituaInvalidos(const string&,
            ValorBase *const)));
367     connect(dialogSubstQuant, SIGNAL(substituaIntervalos(const string&,
            ValorBase *const, const vector<TabelaDeRegistros::Intervalo>&,
            const bool. const bool)), this, SLOT(substituaIntervalos(const
```

```
            string&, ValorBase *const, const vector<TabelaDeRegistros::
            Intervalo>&, const bool. const bool)));
368     connect(dialogSubstQuant, SIGNAL(substituaDestacados(const string&,
            ValorBase *const)), this, SLOT(substituaDestacados(const string&,
            ValorBase *const)));
369
370     connect(dialogSubstQuant, SIGNAL(substituaMissingsDP(const string&,
            const Distribuicao&)), this, SLOT(substituaMissingsDP(const string
            &, const Distribuicao&)));
371     connect(dialogSubstQuant, SIGNAL(substituaTodosDP(const string&, const
            Distribuicao&)), this, SLOT(substituaTodosDP(const string&, const
            Distribuicao&)));
372     connect(dialogSubstQuant, SIGNAL(substituaInvalidosDP(const string&,
            const Distribuicao&)), this, SLOT(substituaInvalidosDP(const
            string&, const Distribuicao&)));
373     connect(dialogSubstQuant, SIGNAL(substituaIntervaloDP(const string&,
            const Distribuicao&, const vector<TabelaDeRegistros::Intervalo>&,
            const bool, const bool)), this, SLOT(substituaIntervaloDP(const
            string&, const Distribuicao&, const vector<TabelaDeRegistros::
            Intervalo>&, const bool, const bool)));
374     connect(dialogSubstQuant, SIGNAL(substituaDestacadosDP(const string&,
            const Distribuicao&)), this, SLOT(substituaDestacadosDP(const
            string&, const Distribuicao&)));
375
376     connect(dialogSubstQuant, SIGNAL(substituaMissingsRLS(const string&,
            const string&, const RegressaoLinearSimples&)), this, SLOT(
            substituaMissingsRLS(const string&, const string&, const
            RegressaoLinearSimples&)));
377     connect(dialogSubstQuant, SIGNAL(substituaTodosRLS(const string&, const
             string&, const RegressaoLinearSimples&)), this, SLOT(
            substituaTodosRLS(const string&, const string&, const
            RegressaoLinearSimples&)));
378     connect(dialogSubstQuant, SIGNAL(substituaInvalidosRLS(const string&,
            const string&, const RegressaoLinearSimples&)), this, SLOT(
            substituaInvalidosRLS(const string&, const string&, const
            RegressaoLinearSimples&)));
379     connect(dialogSubstQuant, SIGNAL(substituaIntervaloRLS(const string&,
            const string&, const RegressaoLinearSimples&, const vector<
            TabelaDeRegistros::Intervalo>&, const bool, const bool)), this,
            SLOT(substituaIntervaloRLS(const string&, const string&, const
            RegressaoLinearSimples&, const vector<TabelaDeRegistros::Intervalo
            >&, const bool, const bool)));
380     connect(dialogSubstQuant, SIGNAL(substituaDestacadosRLS(const string&,
            const string&, const RegressaoLinearSimples&)), this, SLOT(
            substituaDestacadosRLS(const string&, const string&, const
            RegressaoLinearSimples&)));
381
382     dialogSubstQuant->exec();
383     delete dialogSubstQuant;
384 }
385
386
387 void WidgetTabelaEditora::substituaMissings(const string &atributo,
        ValorBase *const novoValor)
388 {
389     TabelaDeRegistros::obterInstancia()->substituaMissings(atributo,
            novoValor);
390
391     vector<string> atributos = vector<string>();
392     atributos.push_back(atributo);
393
394     atualizeEstatisticaColuna(atributos);
395
396     repopularColunas(atributos);
397 }
398
399 void WidgetTabelaEditora::substituaTodos(const string &atributo, ValorBase
        *const novoValor)
400 {
401     TabelaDeRegistros::obterInstancia()->substituaTodos(atributo, novoValor
            );
402
403     vector<string> atributos = vector<string>();
404     atributos.push_back(atributo);
405
406     atualizeEstatisticaColuna(atributos);
407
408     repopularColunas(atributos);
409 }
410
411 void WidgetTabelaEditora::substituaInvalidos(const string &atributo,
        ValorBase *const novoValor)
```

```
412 {
413     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributo))
414     {
415         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Subtituir inválidos só se aplica para atributos
                quantitativos!"));
416         return;
417     }
418
419     TabelaDeRegistros::obterInstancia()->substituaInvalidos(atributo,
            novoValor);
420
421     vector<string> atributos = vector<string>();
422     atributos.push_back(atributo);
423
424     atualizeEstatisticaColuna(atributos);
425
426     repopularColunas(atributos);
427 }
428
429 void WidgetTabelaEditora::substituaDestacados(const string &atributo,
        ValorBase *const novoValor)
430 {
431     TabelaDeRegistros::obterInstancia()->substituaTodosPorIndice(atributo,
            novoValor, _registrosDestacadosPorAtributo[atributo]);
432
433     vector<string> atributos = vector<string>();
434     atributos.push_back(atributo);
435
436     atualizeEstatisticaColuna(atributos);
437
438     repopularColunas(atributos);
439 }
440
441 void WidgetTabelaEditora::substituaIntervalos(const string &atributo,
        ValorBase *const novoValor, const vector<TabelaDeRegistros::Intervalo>
        &intervalos, const bool missingsTmb, const bool invalidosTmb)
442 {
443     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributo))
444     {
445         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Subtituir inválidos só se aplica para atributos
                quantitativos!"));
446         return;
447     }
448
449     TabelaDeRegistros::obterInstancia()->substituaIntervalos(atributo,
            novoValor, intervalos, missingsTmb, invalidosTmb);
450
451     vector<string> atributos = vector<string>();
452     atributos.push_back(atributo);
453
454     atualizeEstatisticaColuna(atributos);
455
456     repopularColunas(atributos);
457 }
458
459 void WidgetTabelaEditora::substituaMissingsDP(const string &atributo, const
        Distribuicao &distribuicao)
460 {
461     TabelaDeRegistros::obterInstancia()->substituaMissingsDPPorIndice(
            atributo, _indicesTodosRegistros, distribuicao);
462
463     vector<string> atributos = vector<string>();
464     atributos.push_back(atributo);
465
466     atualizeEstatisticaColuna(atributos);
467
468     repopularColunas(atributos);
469 }
470
471 void WidgetTabelaEditora::substituaTodosDP(const string &atributo, const
        Distribuicao &distribuicao)
472 {
473     TabelaDeRegistros::obterInstancia()->substituaTodosDPPorIndice(atributo
            , _indicesTodosRegistros, distribuicao);
474
475     vector<string> atributos = vector<string>();
476     atributos.push_back(atributo);
477
478     atualizeEstatisticaColuna(atributos);
479
```

```
480     repopularColunas(atributos);
481 }
482
483 void WidgetTabelaEditora::substituaInvalidosDP(const string &atributo,
        const Distribuicao &distribuicao)
484 {
485     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributo))
486     {
487         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Subtituir inválidos só se aplica para atributos
                quantitativos!"));
488         return;
489     }
490
491     TabelaDeRegistros::obterInstancia()->substituaInvalidosDPPorIndice(
            atributo, _indicesTodosRegistros, distribuicao);
492
493     vector<string> atributos = vector<string>();
494     atributos.push_back(atributo);
495
496     atualizeEstatisticaColuna(atributos);
497
498     repopularColunas(atributos);
499 }
500
501 void WidgetTabelaEditora::substituaIntervaloDP(
502     const string &atributo,
503     const Distribuicao &distribuicao,
504     const vector<TabelaDeRegistros::Intervalo> &intervalos,
505     const bool missingsTmb,
506     const bool invalidosTmb)
507 {
508     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributo))
509     {
510         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Subtituir inválidos só se aplica para atributos
                quantitativos!"));
511         return;
512     }
513
514     TabelaDeRegistros::obterInstancia()->substituaIntervalosDPPorIndice(
            atributo, _indicesTodosRegistros, distribuicao, intervalos,
            missingsTmb, invalidosTmb);
515
516     vector<string> atributos = vector<string>();
517     atributos.push_back(atributo);
518
519     atualizeEstatisticaColuna(atributos);
520
521     repopularColunas(atributos);
522 }
523
524 void WidgetTabelaEditora::substituaDestacadosDP(const string &atributo,
        const Distribuicao &distribuicao)
525 {
526     TabelaDeRegistros::obterInstancia()->substituaTodosDPPorIndice(atributo
            , _registrosDestacadosPorAtributo[atributo], distribuicao);
527
528     vector<string> atributos = vector<string>();
529     atributos.push_back(atributo);
530
531     atualizeEstatisticaColuna(atributos);
532
533     repopularColunas(atributos);
534 }
535
536 void WidgetTabelaEditora::substituaMissingsRLS(const string &atributoDep,
        const string &atributoInd, const RegressaoLinearSimples &regressao)
537 {
538     TabelaDeRegistros::obterInstancia()->substituaMissingsRLSPorIndice(
            atributoInd, atributoDep, _indicesTodosRegistros, regressao);
539
540     vector<string> atributos = vector<string>();
541     atributos.push_back(atributoDep);
542
543     atualizeEstatisticaColuna(atributos);
544
545     repopularColunas(atributos);
546 }
547
548 void WidgetTabelaEditora::substituaTodosRLS(const string &atributoDep,
        const string &atributoInd, const RegressaoLinearSimples &regressao)
```

```
549 {
550     TabelaDeRegistros::obterInstancia()->substituaTodosRLSPorIndice(
            atributoInd, atributoDep, _indicesTodosRegistros, regressao);
551
552     vector<string> atributos = vector<string>();
553     atributos.push_back(atributoDep);
554
555     atualizeEstatisticaColuna(atributos);
556
557     repopularColunas(atributos);
558 }
559
560 void WidgetTabelaEditora::substituaInvalidosRLS(const string &atributoDep,
        const string &atributoInd, const RegressaoLinearSimples &regressao)
561 {
562     TabelaDeRegistros::obterInstancia()->substituaInvalidosRLSPorIndice(
            atributoInd, atributoDep, _indicesTodosRegistros, regressao);
563
564     vector<string> atributos = vector<string>();
565     atributos.push_back(atributoDep);
566
567     atualizeEstatisticaColuna(atributos);
568
569     repopularColunas(atributos);
570 }
571
572 void WidgetTabelaEditora::substituaIntervaloRLS(
573     const string &atributoDep,
574     const string &atributoInd,
575     const RegressaoLinearSimples &regressao,
576     const vector<TabelaDeRegistros::Intervalo> &intervalos,
577     const bool missingsTmb,
578     const bool invalidosTmb)
579 {
580     TabelaDeRegistros::obterInstancia()->substituaIntervalosRLSPorIndice(
            atributoInd, atributoDep, _indicesTodosRegistros, regressao,
            intervalos, missingsTmb, invalidosTmb);
581
582     vector<string> atributos = vector<string>();
583     atributos.push_back(atributoDep);
584
585     atualizeEstatisticaColuna(atributos);
586
587     repopularColunas(atributos);
588 }
589
590 void WidgetTabelaEditora::substituaDestacadosRLS(const string &atributoDep,
        const string &atributoInd, const RegressaoLinearSimples &regressao)
591 {
592
593     TabelaDeRegistros::obterInstancia()->substituaTodosRLSPorIndice(
            atributoInd, atributoDep, _registrosDestacadosPorAtributo[
            atributoDep], regressao);
594
595     vector<string> atributos = vector<string>();
596     atributos.push_back(atributoDep);
597
598     atualizeEstatisticaColuna(atributos);
599
600     repopularColunas(atributos);
601 }
602
603
604 void WidgetTabelaEditora::substituaLocalizados(
605     const string &atributo,
606     const vector<ValorBase *const> &antigosValores,
607     ValorBase *const novoValor,
608     const bool missingsTmb,
609     const bool invalidosTmb)
610 {
611     TabelaDeRegistros::obterInstancia()->substituaLocalizadosPorIndice(
            atributo, _indicesTodosRegistros, antigosValores, novoValor,
            missingsTmb, invalidosTmb);
612
613     vector<string> atributos = vector<string>();
614     atributos.push_back(atributo);
615
616     atualizeEstatisticaColuna(atributos);
617
618     repopularColunas(atributos);
619 }
620
```

```
621 void WidgetTabelaEditora::removerComboBox()
622 {
623     if (_comboBox)
624     {
625         removerComboBox(_comboBox->currentText().toLatin1().data());
626     }
627 }
628
629 void WidgetTabelaEditora::itemComboClick(QString textoSelecionado)
630 {
631     removerComboBox(textoSelecionado.toLatin1().data());
632 }
633
634 void WidgetTabelaEditora::removerComboBox(const string &novoValorStr)
635 {
636     ui.tableViewEdicao->setIndexWidget(*_indiceCombo, NULL);
637
638     const size_t indiceRegistro = _tableViewEdicaoModel->item(_indiceCombo
            ->row(), 0)->text().toLatin1().toInt();
639     const string atributo = _tableViewEdicaoModel->headerData(_indiceCombo
          ->column(), Qt::Horizontal).toString().toLatin1().data();
640
641     if (_comboBox != NULL && _comboBox->currentIndex() == 0) //novoValor ==
            "+"
642     {
643         ui.tableViewEdicao->edit(*_indiceCombo);
644     }
645     else
646     {
647         const string valorAntigo = _tableViewEdicaoModel->item(_indiceCombo
                ->row(), _indiceCombo->column())->text().toLatin1().data();
648
649         if (valorAntigo != novoValorStr)
650         {
651             _tableViewEdicaoModel->blockSignals(true);
652             _tableViewEdicaoModel->item((*_indiceCombo).row(), (*
                    _indiceCombo).column())->setText(QString::fromLatin1(
                    novoValorStr.c_str()));
653             _tableViewEdicaoModel->blockSignals(false);
654
655             ValorBase *novoValor = TabelaDeRegistros::obterInstancia()->
                    gerarValor(atributo, novoValorStr);
656             TabelaDeRegistros::obterInstancia()->atualizeRegistro(
                    indiceRegistro, atributo, novoValor);
657             delete novoValor;
658
659             vector<string> atributos = vector<string>();
660             atributos.push_back(atributo);
661
662             atualizeEstatisticaColuna(atributos);
663         }
664     }
665
666     delete _comboBox;
667     _comboBox = NULL;
668
669     delete _indiceCombo;
670     _indiceCombo = NULL;
671 }
672
673 void WidgetTabelaEditora::celulaClick(const QModelIndex &indice)
674 {
675     if (indice.row() == _linhaClick && indice.column() == _colunaClick)
676     {
677         return;
678     }
679
680     _linhaClick = indice.row();
681     _colunaClick = indice.column();
682
683     removerComboBox();
684 }
685
686 void WidgetTabelaEditora::celulaClickDuplo(const QModelIndex &indice)
687 {
688     const string atributo = _tableViewEdicaoModel->headerData(indice.column
            (), Qt::Horizontal).toString().toLatin1().data();
689
690     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributo))
691     {
692         const string valorAnterior = ui.tableViewEdicao->model()->data(
                indice).toString().toLatin1().data();
```

```
693
694             _comboBox = new QComboBox();
695             _indiceCombo = new QModelIndex(indice); //deletado em ~
                    WidgetTabelaEditora e removerComboBox
696
697             vector<string> categoriasAtributoQualitativo = vector<string>();
698             categoriasAtributoQualitativo.push_back("+");
699             ColetorEstatisticas::obterInstancia()->todasCategoriasAtributoQual(
                    atributo, categoriasAtributoQualitativo);
700
701             Populador::populeComboBox(_comboBox, categoriasAtributoQualitativo)
                    ;
702
703             _comboBox->setCurrentText(QString::fromLatin1(valorAnterior.c_str()
                    ));
704
705             ui.tableViewEdicao->setIndexWidget(*_indiceCombo, _comboBox);
706
707             connect(_comboBox, SIGNAL(activated(QString)), this, SLOT(
                    itemComboClick(QString)));
708         }
709     }
710
711     void WidgetTabelaEditora::celulaAlterada(QStandardItem *item)
712     {
713         const int linhaIndice = item->index().row();
714         const int colunaIndice = item->index().column();
715
716         const size_t indiceRegistro = _tableViewEdicaoModel->item(linhaIndice,
                0)->text().toLatin1().toInt();
717         const string atributo = _tableViewEdicaoModel->headerData(colunaIndice,
                Qt::Horizontal).toString().toLatin1().data();
718         const size_t indiceAtributo = TabelaDeRegistros::obterInstancia()->
                indiceAtributo(atributo);
719
720         const string novoValorStr = item->text().toLatin1().data();
721
722         bool ok = false;
723         ValorBase *novoValor;
724         double novoValorDbl;
725
726         if (TabelaDeRegistros::obterInstancia()->ehQuant(atributo))
727         {
728             novoValorDbl = Utilidades::stdStrToDbl(novoValorStr, true, true, &
                    ok);
729
730             if (!ok)
731             {
732                 QMessageBox msgQuestion(QMessageBox::Question, "Aviso", QString
                        ::fromLatin1("O valor inserido não é quantitativo. Ex.:
                        \"1,57\", \"10000,57\", \"100\". \nDeseja editar mesmo
                        assim?"), QMessageBox::Yes|QMessageBox::No, this);
733                 msgQuestion.button(QMessageBox::Yes)->setText(QString::
                        fromLatin1("Sim"));
734                 msgQuestion.button(QMessageBox::No)->setText(QString::
                        fromLatin1("Não"));
735
736                 if (QMessageBox::No == msgQuestion.exec())
737                 {
738                     const Registro &r = TabelaDeRegistros::obterInstancia()->
                            registro(indiceRegistro);
739                     const string valorAntigoStr = r.retorneValor(indiceAtributo
                            )->valorStr();
740
741                     _tableViewEdicaoModel->blockSignals(true);
742                     item->setText(QString::fromLatin1(valorAntigoStr.c_str()));
743                     _tableViewEdicaoModel->blockSignals(false);
744
745                     return;
746                 }
747             }
748         }
749
750         if (ok)
751         {
752             novoValor = TabelaDeRegistros::gerarValorQuant(novoValorDbl);
753         }
754         else
755         {
756             novoValor = TabelaDeRegistros::obterInstancia()->gerarValor(
                    atributo, novoValorStr);
757         }
```

```
758
759     //essas tres linhas é pra setar celula com o que o usuario digitou, mas
            de forma padronizada
760     _tableViewEdicaoModel->blockSignals(true);
761     item->setText(QString::fromLatin1(novoValor->valorStr().c_str()));
762     _tableViewEdicaoModel->blockSignals(false);
763
764     TabelaDeRegistros::obterInstancia()->atualizeRegistro(indiceRegistro,
            atributo, novoValor);
765     delete novoValor;
766
767     vector<string> atributos = vector<string>();
768     atributos.push_back(atributo);
769
770     atualizeEstatisticaColuna(atributos);
771 }
772
773 void WidgetTabelaEditora::removerColuna()
774 {
775     mostrarTodos();
776
777     if (TabelaDeRegistros::obterInstancia()->totalAtributos() <= 1)
778     {
779         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Não é possível remover o último atributo!"));
780         return;
781     }
782
783     const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
784     if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
            contem os ids
785     {
786         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Selecione uma coluna!"));
787         return;
788     }
789
790     const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
            Qt::Horizontal).toString().toLatin1().data();
791
792     TabelaDeRegistros::obterInstancia()->removerAtributo(atributo);
793     ColetorEstatisticas::obterInstancia()->removerAtributo(atributo);
794
795     _tableViewEdicaoModel->removeColumn(indiceColuna);
796 }
797
798 void WidgetTabelaEditora::removerLinha(const bool removerRegistro)
799 {
800     removerComboBox();
801
802     if (TabelaDeRegistros::obterInstancia()->totalRegistros() == 1)
803     {
804         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Não é possível remover o último registro!"));
805         return;
806     }
807
808     const int linha = ui.tableViewEdicao->currentIndex().row();
809
810     if (linha == -1)
811     {
812         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Selecione uma linha!"));
813         return;
814     }
815
816     const size_t indiceRegistro = _tableViewEdicaoModel->item(linha, 0)->
            text().toLatin1().toInt();
817
818     if (removerRegistro)
819     {
820         TabelaDeRegistros::obterInstancia()->removaRegistro(indiceRegistro)
                ;
821     }
822
823     _tableViewEdicaoModel->removeRow(linha);
824     _totalLinhas--;
825
826     vector<size_t>::const_iterator cit = std::find(_indicesTodosRegistros.
            cbegin(), _indicesTodosRegistros.cend(), indiceRegistro);
827     if (cit != _indicesTodosRegistros.cend())
828     {
```

```
829             _indicesTodosRegistros.erase(cit);
830         }
831
832         atualizeEstatisticaColuna(TabelaDeRegistros::obterInstancia()->
                atributos());
833     }
834
835     void WidgetTabelaEditora::removerLinhas(const vector<size_t> &
            indiceRegistros, const bool removerRegistro)
836     {
837         if (indiceRegistros.empty())
838         {
839             return;
840         }
841
842         const size_t totalLinhasRemover = indiceRegistros.size();
843
844         QProgressDialog *progress = new QProgressDialog(QString::fromLatin1("
                Removendo linhas..."), "", 0, 100, this);
845         progress->setWindowTitle(QString::fromLatin1("Aguarde"));
846         progress->setWindowModality(Qt::ApplicationModal);
847         progress->setWindowFlags(progress->windowFlags() & ~Qt::
                WindowContextHelpButtonHint & ~Qt::WindowCloseButtonHint | Qt::
                MSWindowsFixedSizeDialogHint);
848         progress->setCancelButton(NULL);
849         progress->setValue(0);
850         progress->show();
851
852         const int taxaNotificacao = totalLinhasRemover <= 10 ? 10 :
                totalLinhasRemover / 10;
853         size_t i = 0;
854
855         mostrarTodos();
856
857         ui.tableViewEdicao->blockSignals(true);
858         ui.tableViewEdicao->setUpdatesEnabled(false);
859         _tableViewEdicaoModel->blockSignals(true);
860
861         //invertido para nao afetar o indice
862         for (int linha = _tableViewEdicaoModel->rowCount() - 1; linha >= 0;
                linha--)
863         {
864             const size_t indiceRegistro = _tableViewEdicaoModel->item(linha, 0)
                    ->text().toLatin1().toInt();
865
866             if (std::find(indiceRegistros.begin(), indiceRegistros.end(),
                    indiceRegistro) != indiceRegistros.end())
867             {
868                 _tableViewEdicaoModel->removeRow(linha);
869                 _totalLinhas--;
870
871                 if (removerRegistro)
872                 {
873                     TabelaDeRegistros::obterInstancia()->removaRegistro(
                            indiceRegistro);
874                 }
875
876                 vector<size_t>::const_iterator cit = std::find(
                        _indicesTodosRegistros.cbegin(), _indicesTodosRegistros.
                        cend(), indiceRegistro);
877                 if (cit != _indicesTodosRegistros.cend())
878                 {
879                     _indicesTodosRegistros.erase(cit);
880                 }
881
882                 i++;
883                 if ((i % taxaNotificacao) == 0)
884                 {
885                     progress->setValue(min((int)(i * 1.0 / totalLinhasRemover *
                            100), 99));
886                     QCoreApplication::processEvents(QEventLoop::
                            ExcludeUserInputEvents);
887                 }
888             }
889         }
890
891         _tableViewEdicaoModel->setRowCount(_totalLinhas);
892
893         ui.tableViewEdicao->blockSignals(false);
894         ui.tableViewEdicao->setUpdatesEnabled(true);
895         _tableViewEdicaoModel->blockSignals(false);
896         emit _tableViewEdicaoModel->layoutChanged();
```

```
897
898         delete progress;
899
900         atualizeEstatisticaColuna(TabelaDeRegistros::obterInstancia()->
                atributos());
901     }
902
903     void WidgetTabelaEditora::removerLinhasDestacadasDoAtributo(const bool
            removerRegistro)
904     {
905         const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
906         if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
                contem os ids
907         {
908             QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma coluna!"));
909             return;
910         }
911
912         const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
                Qt::Horizontal).toString().toLatin1().data();
913
914         if (_registrosDestacadosPorAtributo[atributo].size() <= 0)
915         {
916             QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Não há valores destacados na coluna"));
917             return;
918         }
919
920         removerLinhas(_registrosDestacadosPorAtributo[atributo]);
921     }
922
923     void WidgetTabelaEditora::removerTodasLinhas()
924     {
925         const size_t totalLinhasRemover = _tableViewEdicaoModel->rowCount();
926
927         QProgressDialog *progress = new QProgressDialog(QString::fromLatin1("
                Removendo linhas..."), "", 0, 100, this);
928         progress->setWindowTitle(QString::fromLatin1("Aguarde"));
929         progress->setWindowModality(Qt::ApplicationModal);
930         progress->setWindowFlags(progress->windowFlags() & ~Qt::
                WindowContextHelpButtonHint & ~Qt::WindowCloseButtonHint | Qt::
                MSWindowsFixedSizeDialogHint);
931         progress->setCancelButton(NULL);
932         progress->setValue(0);
933         progress->show();
934
935         const int taxaNotificacao = totalLinhasRemover <= 10 ? 10 :
                totalLinhasRemover / 10;
936         size_t i = 0;
937
938         removerComboBox();
939
940         ui.tableViewEdicao->setUpdatesEnabled(false);
941         ui.tableViewEdicao->blockSignals(true);
942         _tableViewEdicaoModel->blockSignals(true);
943
944         while (_tableViewEdicaoModel->rowCount() > 0 && TabelaDeRegistros::
                obterInstancia()->totalRegistros() > 1)
945         {
946             const size_t indiceRegistro = _tableViewEdicaoModel->item(0, 0)->
                    text().toLatin1().toInt();
947
948             TabelaDeRegistros::obterInstancia()->removaRegistro(indiceRegistro)
                    ;
949
950             _tableViewEdicaoModel->removeRow(0);
951             _totalLinhas--;
952
953             vector<size_t>::const_iterator cit = std::find(
                    _indicesTodosRegistros.cbegin(), _indicesTodosRegistros.cend()
                    , indiceRegistro);
954             if (cit != _indicesTodosRegistros.cend())
955             {
956                 _indicesTodosRegistros.erase(cit);
957             }
958
959             i++;
960             if ((i % taxaNotificacao) == 0)
961             {
962                 progress->setValue(min((int)(i * 1.0 / totalLinhasRemover *
                        100), 99));
```

```
963                 QCoreApplication::processEvents(QEventLoop::
                        ExcludeUserInputEvents);
964             }
965         }
966
967         ui.tableViewEdicao->setUpdatesEnabled(true);
968         ui.tableViewEdicao->blockSignals(false);
969         _tableViewEdicaoModel->blockSignals(false);
970         emit _tableViewEdicaoModel->layoutChanged();
971
972         if (TabelaDeRegistros::obterInstancia()->totalRegistros() == 1)
973         {
974             QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Não é possível remover o último registro!"));
975         }
976
977         delete progress;
978
979         atualizeEstatisticaColuna(TabelaDeRegistros::obterInstancia()->
                atributos());
980 }
981
982 void WidgetTabelaEditora::inserirLinha()
983 {
984         mostrarTodos();
985
986         const size_t indiceNovoRegistro = TabelaDeRegistros::obterInstancia()->
                novoRegistro();
987         const Registro &r = TabelaDeRegistros::obterInstancia()->registro(
                indiceNovoRegistro);
988
989         const size_t indiceNovaLinha = _tableViewEdicaoModel->rowCount();
990         _tableViewEdicaoModel->insertRow(_tableViewEdicaoModel->rowCount());
991         _totalLinhas++;
992
993         _indicesTodosRegistros.push_back(indiceNovoRegistro);
994
995         vector<string> registroVetor = r.valoresString();
996         registroVetor.insert(registroVetor.begin(), Utilidades::dblToStdStr(
                indiceNovoRegistro, 0));
997
998         Populador::populeLinhaTabela(registroVetor, indiceNovaLinha, ui.
                tableViewEdicao);
999
1000        atualizeEstatisticaColuna(TabelaDeRegistros::obterInstancia()->
                atributos());
1001
1002        ui.tableViewEdicao->setCurrentIndex(_tableViewEdicaoModel->index(
                indiceNovaLinha, 1));
1003 }
1004
1005 void WidgetTabelaEditora::inserirColunaDialog()
1006 {
1007         focarNosRegistros(vector<size_t>(), true);
1008
1009         DialogNovaColuna *dialogNovaColuna = new DialogNovaColuna(this);
1010         connect(dialogNovaColuna, SIGNAL(inserirColuna(const string&, const
                TipoDado_AT)), this, SLOT(inserirColuna(const string&, const
                TipoDado_AT)));
1011
1012         dialogNovaColuna->exec();
1013         delete dialogNovaColuna;
1014 }
1015
1016 void WidgetTabelaEditora::inserirColuna(const string &atributo, const
        TipoDado_AT tipo)
1017 {
1018         TabelaDeRegistros::obterInstancia()->adicionarAtributo(atributo, tipo);
1019
1020         vector<string> atributos = vector<string>();
1021         atributos.push_back(atributo);
1022         atualizeEstatisticaColuna(atributos);
1023
1024         _tableViewEdicaoModel->setColumnCount(_tableViewEdicaoModel->
                columnCount() + 1);
1025
1026         map<size_t, string> colunaStr = map<size_t, string>();
1027         TabelaDeRegistros::obterInstancia()->valoresAtributoStr(atributo,
                colunaStr);
1028
1029         Populador::populeColunaTabelaEditora(colunaStr, _tableViewEdicaoModel->
                columnCount() - 1, ui.tableViewEdicao, atributo);
```

```
1030 }
1031
1032 void WidgetTabelaEditora::repopularColunas(const vector<string> &atributos)
1033 {
1034     map<size_t, string> colunaStr = map<size_t, string>();
1035
1036     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
1037          citAtributo != atributos.cend();
1038          citAtributo++)
1039     {
1040         const string &atributo = *citAtributo;
1041
1042         colunaStr.clear();
1043         TabelaDeRegistros::obterInstancia()->valoresAtributoStrPorIndice(
                 atributo, _indicesTodosRegistros, colunaStr);
1044
1045         const size_t indiceAtributo = TabelaDeRegistros::obterInstancia()->
                 indiceAtributo(*citAtributo);
1046         Populador::populeColunaTabelaEditora(colunaStr, indiceAtributo + 1,
                  ui.tableViewEdicao); //+1 pois a pprimeira coluna tem os ids
1047     }
1048 }
1049
1050 void WidgetTabelaEditora::renomearColunaDialog()
1051 {
1052     removerComboBox();
1053
1054     const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
1055     if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
            contem os ids
1056     {
1057         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                fromLatin1("Selecione uma coluna!"));
1058         return;
1059     }
1060     const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
             Qt::Horizontal).toString().toLatin1().data();
1061
1062     DialogRenomear *dialogRenomear = new DialogRenomear(atributo, this); //
            deletado logo abaixo
1063     connect(dialogRenomear, SIGNAL(renomear(const string&, const string&)),
             this, SLOT(renomearColuna(const string&, const string&)));
1064
1065     dialogRenomear->exec();
1066     delete dialogRenomear;
1067 }
1068
1069 void WidgetTabelaEditora::renomearColuna(const string &nomeAntigo, const
         string &nomeNovo)
1070 {
1071     TabelaDeRegistros::obterInstancia()->renomerAtributo(nomeAntigo,
            nomeNovo);
1072     ColetorEstatisticas::obterInstancia()->renomerAtributo(nomeAntigo,
            nomeNovo);
1073
1074     const size_t indiceAtributo = TabelaDeRegistros::obterInstancia()->
            indiceAtributo(nomeNovo);
1075
1076     QStandardItemModel *model = (QStandardItemModel*)ui.tableViewEdicao->
            model();
1077     model->setHorizontalHeaderItem(indiceAtributo + 1, new QStandardItem(
            QString::fromLatin1(nomeNovo.c_str()))); //+1 pois a primeira
            coluna tem os ids
1078
1079     //ui.tableViewEdicao->resizeColumnsToContents(); lento demais
1080
1081     //todo _exportarBDAct->setEnabled(TabelaDeRegistros::obterInstancia()->
            nenhumAtributoAlteradoRemovido());
1082 }
1083
1084 void WidgetTabelaEditora::focarNosRegistros(const vector<size_t> &registros
         , const bool mostrarTodos) const
1085 {
1086     const int totalLinhas = _tableViewEdicaoModel->rowCount();
1087
1088     if (mostrarTodos)
1089     {
1090         for (int linha = 0; linha < totalLinhas; linha++)
1091         {
1092             ui.tableViewEdicao->setRowHidden(linha, false);
1093         }
1094     }
```

```
1095     else
1096     {
1097         for (int linha = 0; linha < totalLinhas; linha++)
1098         {
1099             const size_t indiceRegistroLinha = _tableViewEdicaoModel->item(
                     linha, 0)->text().toLatin1().toInt();
1100             ui.tableViewEdicao->setRowHidden(linha, std::find(registros.
                     begin(), registros.end(), indiceRegistroLinha) ==
                     registros.end());
1101         }
1102     }
1103 }
1104
1105 void WidgetTabelaEditora::mostrarTodos()
1106 {
1107     removerComboBox();
1108     focarNosRegistros(vector<size_t>(), true);
1109 }
1110
1111 void WidgetTabelaEditora::atualizeEstatisticaColuna(const vector<string> &
         atributos)
1112 {
1113     habilitarEventos(false);
1114
1115     ThreadWorkerColetarEstatisticas *workerEstatistica = new
           ThreadWorkerColetarEstatisticas(atributos, this);
1116     _threadController.setarWorker(workerEstatistica);
1117     _threadController.iniciarExecucaoWorker(this);
1118 }
1119
1120 void WidgetTabelaEditora::resizeEvent(QResizeEvent *evento)
1121 {
1122     QWidget::resizeEvent(evento);
1123
1124     const QSize &size = evento->size();
1125
1126     _menu->resize(size.width(), _menu->height());
1127 }
1128
1129
1130 void WidgetTabelaEditora::onUpdateProgresso(int *porcentagem)
1131 {
1132     ui.progressBar->setValue(*porcentagem);
1133     delete porcentagem;
1134 }
1135
1136 void WidgetTabelaEditora::fimColetaEstatisticas(const vector<string> *
         atributos)
1137 {
1138     ui.progressBar->setValue(0);
1139     delete atributos;
1140
1141     habilitarEventos(true);
1142 }
1143
1144 void WidgetTabelaEditora::resetarDestaque()
1145 {
1146     const int totalLinhas = _tableViewEdicaoModel->rowCount();
1147     const int totalColunas = _tableViewEdicaoModel->columnCount();
1148
1149     ui.tableViewEdicao->setUpdatesEnabled(false);
1150     ui.tableViewEdicao->blockSignals(true);
1151     _tableViewEdicaoModel->blockSignals(true);
1152
1153     for (int linha = 0; linha < totalLinhas; linha++)
1154     {
1155         for (int coluna = 1; coluna < totalColunas; coluna++)
1156         {
1157             _tableViewEdicaoModel->item(linha, coluna)->setBackground(Qt::
                     white);
1158         }
1159     }
1160
1161     ui.tableViewEdicao->setUpdatesEnabled(true);
1162     ui.tableViewEdicao->blockSignals(false);
1163     _tableViewEdicaoModel->blockSignals(false);
1164     emit _tableViewEdicaoModel->layoutChanged();
1165 }
1166 void WidgetTabelaEditora::destacar(const map<string, vector<size_t>> &
         indicesPorAtributo)
1167 {
1168     resetarDestaque();
```

```
1169
1170        _registrosDestacadosPorAtributo = indicesPorAtributo;
1171
1172        const int totalLinhas = _tableViewEdicaoModel->rowCount();
1173        const int totalColunas = _tableViewEdicaoModel->columnCount();
1174
1175        map<size_t, int> mapIdParaLinha = map<size_t, int>();
1176
1177        vector<size_t> colunaIds = vector<size_t>();
1178        for (int linha = 0; linha < totalLinhas; linha++)
1179        {
1180            const size_t indiceRegistro = _tableViewEdicaoModel->item(linha, 0)
                    ->text().toLatin1().toInt();
1181            mapIdParaLinha[indiceRegistro] = linha;
1182        }
1183
1184        ui.tableViewEdicao->setUpdatesEnabled(false);
1185        ui.tableViewEdicao->blockSignals(true);
1186        _tableViewEdicaoModel->blockSignals(true);
1187
1188        for (map<string, vector<size_t>>::const_iterator citAtribIdxs =
                _registrosDestacadosPorAtributo.cbegin();
1189            citAtribIdxs != _registrosDestacadosPorAtributo.cend();
1190            citAtribIdxs++)
1191        {
1192            const string &atributo = citAtribIdxs->first;
1193            const vector<size_t> &idxsAtributo = citAtribIdxs->second;
1194
1195            const size_t indiceAtributo = TabelaDeRegistros::obterInstancia()->
                    indiceAtributo(atributo);
1196
1197            for (vector<size_t>::const_iterator citIdxReg = idxsAtributo.cbegin
                    ();
1198                citIdxReg != idxsAtributo.cend();
1199                citIdxReg++)
1200            {
1201                const size_t indiceRegistro = *citIdxReg;
1202
1203                const int linha = mapIdParaLinha[indiceRegistro];
1204                const int coluna = indiceAtributo + 1;//+1 pois a primeira
                        coluna tem os ids
1205
1206                _tableViewEdicaoModel->item(linha, coluna)->setBackground(
                        QColor(255, 100, 100));
1207            }
1208        }
1209
1210        ui.tableViewEdicao->setUpdatesEnabled(true);
1211        ui.tableViewEdicao->blockSignals(false);
1212        _tableViewEdicaoModel->blockSignals(false);
1213        emit _tableViewEdicaoModel->layoutChanged();
1214    }
1215
1216    void WidgetTabelaEditora::setBarraMenuVisivel(const bool visivel)
1217    {
1218        _menu->setVisible(visivel);
1219        ui.verticalSpacer->changeSize(20, visivel ? 35 : 0);
1220    }
1221
1222    void WidgetTabelaEditora::aposIdentificarRegistro(const vector<size_t> &
            registrosIdentificados) const
1223    {
1224        string msg;
1225
1226        if (registrosIdentificados.empty())
1227        {
1228            msg = "Nenhum registro localizado.";
1229        }
1230        else
1231        {
1232            focarNosRegistros(registrosIdentificados, false);
1233            msg = Utilidades::dblToStdStr(registrosIdentificados.size(), 0)
1234                + " registro(s) localizado(s).\n";
1235        }
1236
1237        QMessageBox::information(NULL, "", QString::fromLatin1(msg.c_str()));
1238    }
1239
1240    void WidgetTabelaEditora::aposRemoverRegistro(const vector<size_t> &
            registrosRemovidos)
1241    {
1242        removerLinhas(registrosRemovidos, false);
```

```
1243
1244        const string msg = Utilidades::dblToStdStr(registrosRemovidos.size(),
                0)
1245            + " registro(s) removido(s).\n";
1246
1247        QMessageBox::information(NULL, "", QString::fromLatin1(msg.c_str()));
1248    }
1249
1250    bool WidgetTabelaEditora::atributoSelecionado(string &atributo)
1251    {
1252        const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
1253        if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
                contem os ids
1254        {
1255            QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma coluna!"));
1256            return false;
1257        }
1258
1259        atributo = _tableViewEdicaoModel->headerData(indiceColuna, Qt::
                Horizontal).toString().toLatin1().data();
1260        return true;
1261    }
1262
1263    void WidgetTabelaEditora::gerarValorDPDialog()
1264    {
1265        removerComboBox();
1266
1267        const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
1268        if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
                contem os ids
1269        {
1270            QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma coluna!"));
1271            return;
1272        }
1273        const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
                 Qt::Horizontal).toString().toLatin1().data();
1274
1275        if (!TabelaDeRegistros::obterInstancia()->ehQuant(atributo))
1276        {
1277            QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma coluna quantitativa!"));
1278            return;
1279        }
1280
1281        vector<size_t> indicesValidos = vector<size_t>();
1282        TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
                _registrosDestacadosPorAtributo[atributo], indicesValidos);
1283
1284        DialogEscolherDistribuicao *dialogEscolherDistribuicao = new
                DialogEscolherDistribuicao(atributo, indicesValidos,
                WidgetEscolherDP::Opcao::ESCOLHER, this);
1285        connect(dialogEscolherDistribuicao->_widgetEscolherDP, SIGNAL(
                distribuicoesEscolhidas(const vector<string>&, const vector<const
                Distribuicao>&)), this, SLOT(popularLinhaDP(const vector<string>&,
                 const vector<const Distribuicao>&)));
1286
1287        dialogEscolherDistribuicao->exec();
1288        delete dialogEscolherDistribuicao;
1289    }
1290
1291    void WidgetTabelaEditora::gerarLinhaDPDialog()
1292    {
1293        removerComboBox();
1294        const int linha = ui.tableViewEdicao->currentIndex().row();
1295        if (linha == -1)
1296        {
1297            QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma linha!"));
1298            return;
1299        }
1300
1301        const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
                ()->atributosQuantitativos();
1302
1303        vector<vector<size_t>> registrosDestacadosPorAtributoVetor = vector<
                vector<size_t>>();
1304        vector<vector<size_t>> indicesValidosPorAtributo = vector<vector<size_t
                >>();
1305
1306        for (vector<string>::const_iterator citAtributo = atributosQuant.cbegin
```

```
            ();
1307            citAtributo != atributosQuant.cend();
1308            citAtributo++)
1309        {
1310            const string &atributo = *citAtributo;
1311            registrosDestacadosPorAtributoVetor.push_back(
                    _registrosDestacadosPorAtributo[atributo]);
1312        }
1313
1314        TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
                registrosDestacadosPorAtributoVetor, indicesValidosPorAtributo);
1315
1316        DialogEscolherDistribuicao *dialogEscolherDistribuicao = new
                DialogEscolherDistribuicao(atributosQuant,
                indicesValidosPorAtributo, WidgetEscolherDP::Opcao::ESCOLHER, this
                );
1317        connect(dialogEscolherDistribuicao->_widgetEscolherDP, SIGNAL(
                distribuicoesEscolhidas(const vector<string>&, const vector<const
                Distribuicao>&)), this, SLOT(popularLinhaDP(const vector<string>&,
                 const vector<const Distribuicao>&)));
1318
1319        dialogEscolherDistribuicao->exec();
1320        delete dialogEscolherDistribuicao;
1321    }
1322
1323    void WidgetTabelaEditora::gerarColunaDPDialog()
1324    {
1325        const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
1326        if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
                contem os ids
1327        {
1328            QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma coluna!"));
1329            return;
1330        }
1331        const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
                 Qt::Horizontal).toString().toLatin1().data();
1332
1333        if (!TabelaDeRegistros::obterInstancia()->ehQuant(atributo))
1334        {
1335            QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma coluna quantitativa!"));
1336            return;
1337        }
1338
1339        vector<size_t> indicesValidos = vector<size_t>();
1340        TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
                _registrosDestacadosPorAtributo[atributo], indicesValidos);
1341
1342        DialogEscolherDistribuicao *dialogEscolherDistribuicao = new
                DialogEscolherDistribuicao(atributo, indicesValidos,
                WidgetEscolherDP::Opcao::ESCOLHER, this);
1343        connect(dialogEscolherDistribuicao->_widgetEscolherDP, SIGNAL(
                distribuicoesEscolhidas(const vector<string>&, const vector<const
                Distribuicao>&)), this, SLOT(popularColunaDP(const vector<string
                >&, const vector<const Distribuicao>&)));
1344
1345        dialogEscolherDistribuicao->exec();
1346        delete dialogEscolherDistribuicao;
1347    }
1348
1349    void WidgetTabelaEditora::gerarValorRLSDialog()
1350    {
1351        removerComboBox();
1352
1353        const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
1354        if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
                contem os ids
1355        {
1356            QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma coluna!"));
1357            return;
1358        }
1359        const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
                 Qt::Horizontal).toString().toLatin1().data();
1360
1361        if (!TabelaDeRegistros::obterInstancia()->ehQuant(atributo))
1362        {
1363            QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma coluna quantitativa!"));
1364            return;
1365        }
```

```
1366
1367        const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
                ()->atributosQuantitativos();
1368
1369        vector<vector<size_t>> registrosDestacadosPorAtributoVetor = vector<
                vector<size_t>>();
1370        vector<vector<size_t>> indicesValidosPorAtributo = vector<vector<size_t
                >>();
1371
1372        for (vector<string>::const_iterator citAtributo = atributosQuant.cbegin
                ();
1373             citAtributo != atributosQuant.cend();
1374             citAtributo++)
1375        {
1376            const string &atributo = *citAtributo;
1377            registrosDestacadosPorAtributoVetor.push_back(
                    _registrosDestacadosPorAtributo[atributo]);
1378        }
1379
1380        TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
                registrosDestacadosPorAtributoVetor, indicesValidosPorAtributo);
1381
1382        DialogEscolherRLS *dialogEscolherRLS = new DialogEscolherRLS(atributo,
                atributosQuant, indicesValidosPorAtributo, this);
1383        connect(dialogEscolherRLS, SIGNAL(regressoesEscolhidas(const vector<
                string>&, const vector<string>&, const vector<const
                RegressaoLinearSimples>&)), this, SLOT(popularLinhaRLS(const
                vector<string>&, const vector<string>&, const vector<const
                RegressaoLinearSimples>&)));
1384
1385        dialogEscolherRLS->exec();
1386        delete dialogEscolherRLS;
1387    }
1388
1389    void WidgetTabelaEditora::gerarLinhaRLSDialog()
1390    {
1391        removerComboBox();
1392        const int linha = ui.tableViewEdicao->currentIndex().row();
1393        if (linha == -1)
1394        {
1395            QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                    fromLatin1("Selecione uma linha!"));
1396            return;
1397        }
1398
1399        const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
                ()->atributosQuantitativos();
1400
1401        vector<vector<size_t>> registrosDestacadosPorAtributoVetor = vector<
                vector<size_t>>();
1402        vector<vector<size_t>> indicesValidosPorAtributo = vector<vector<size_t
                >>();
1403
1404        for (vector<string>::const_iterator citAtributo = atributosQuant.cbegin
                ();
1405             citAtributo != atributosQuant.cend();
1406             citAtributo++)
1407        {
1408            const string &atributo = *citAtributo;
1409            registrosDestacadosPorAtributoVetor.push_back(
                    _registrosDestacadosPorAtributo[atributo]);
1410        }
1411
1412        TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
                registrosDestacadosPorAtributoVetor, indicesValidosPorAtributo);
1413
1414        DialogEscolherRLS *dialogEscolherRLS = new DialogEscolherRLS(
                atributosQuant, atributosQuant, indicesValidosPorAtributo, this);
1415        connect(dialogEscolherRLS, SIGNAL(regressoesEscolhidas(const vector<
                string>&, const vector<string>&, const vector<const
                RegressaoLinearSimples>&)), this, SLOT(popularLinhaRLS(const
                vector<string>&, const vector<string>&, const vector<const
                RegressaoLinearSimples>&)));
1416
1417        dialogEscolherRLS->exec();
1418        delete dialogEscolherRLS;
1419    }
1420    void WidgetTabelaEditora::gerarColunaRLSDialog()
1421    {
1422        const int indiceColuna = ui.tableViewEdicao->currentIndex().column();
1423        if (indiceColuna < 1) //nao pode ser 0, pois a coluna 0 (primeira)
                contem os ids
```

```
1424     {
1425         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                 fromLatin1("Selecione uma coluna!"));
1426         return;
1427     }
1428     const string atributo = _tableViewEdicaoModel->headerData(indiceColuna,
             Qt::Horizontal).toString().toLatin1().data();
1429
1430     if (!TabelaDeRegistros::obterInstancia()->ehQuant(atributo))
1431     {
1432         QMessageBox::warning(NULL, QString::fromLatin1("Aviso"), QString::
                 fromLatin1("Selecione uma coluna quantitativa!"));
1433         return;
1434     }
1435
1436     const vector<string> atributosQuant = TabelaDeRegistros::obterInstancia
             ()->atributosQuantitativos();
1437
1438     vector<vector<size_t>> registrosDestacadosPorAtributoVetor = vector<
             vector<size_t>>();
1439     vector<vector<size_t>> indicesValidosPorAtributo = vector<vector<size_t
             >>();
1440
1441     for (vector<string>::const_iterator citAtributo = atributosQuant.cbegin
             ();
1442         citAtributo != atributosQuant.cend();
1443         citAtributo++)
1444     {
1445         const string &atributo = *citAtributo;
1446         registrosDestacadosPorAtributoVetor.push_back(
                 _registrosDestacadosPorAtributo[atributo]);
1447     }
1448
1449     TabelaDeRegistros::obterInstancia()->indiceRegistrosComplementares(
             registrosDestacadosPorAtributoVetor, indicesValidosPorAtributo);
1450
1451     DialogEscolherRLS *dialogEscolherRLS = new DialogEscolherRLS(atributo,
             atributosQuant, indicesValidosPorAtributo, this);
1452     connect(dialogEscolherRLS, SIGNAL(regressoesEscolhidas(const vector<
             string>&, const vector<string>&, const vector<const
             RegressaoLinearSimples>&)), this, SLOT(popularColunaRLS(const
             vector<string>&, const vector<string>&, const vector<const
             RegressaoLinearSimples>&)));
1453
1454     dialogEscolherRLS->exec();
1455     delete dialogEscolherRLS;
1456 }
1457
1458
1459 void WidgetTabelaEditora::popularLinhaDP(const vector<string> &atributos,
         const vector<const Distribuicao> &distribuicoes)
1460 {
1461     const int linha = ui.tableViewEdicao->currentIndex().row();
1462     const int coluna = ui.tableViewEdicao->currentIndex().column();
1463
1464     const size_t indiceRegistro = _tableViewEdicaoModel->item(linha, 0)->
             text().toLatin1().toInt();
1465
1466     TabelaDeRegistros::obterInstancia()->popularRegistroDP(indiceRegistro,
             atributos, distribuicoes);
1467
1468     const Registro &r = TabelaDeRegistros::obterInstancia()->registro(
             indiceRegistro);
1469     vector<string> registroVetor = r.valoresString();
1470     registroVetor.insert(registroVetor.begin(), Utilidades::dblToStdStr(
             indiceRegistro, 0));
1471     Populador::populeLinhaTabela(registroVetor, linha, ui.tableViewEdicao);
1472
1473     atualizeEstatisticaColuna(atributos);
1474
1475     ui.tableViewEdicao->setCurrentIndex(_tableViewEdicaoModel->index(linha,
             coluna));
1476 }
1477
1478 void WidgetTabelaEditora::popularColunaDP(const vector<string> &atributos,
         const vector<const Distribuicao> &distribuicoes)
1479 {
1480     const int linha = ui.tableViewEdicao->currentIndex().row();
1481
1482     const size_t indiceRegistro = _tableViewEdicaoModel->item(linha, 0)->
             text().toLatin1().toInt();
1483
```

```
1484      TabelaDeRegistros::obterInstancia()->popularColunasDP(atributos,
              distribuicoes);
1485
1486      atualizeEstatisticaColuna(atributos);
1487
1488      repopularColunas(atributos);
1489  }
1490
1491  void WidgetTabelaEditora::popularLinhaRLS(
1492      const vector<string> &atributosIndep,
1493      const vector<string> &atributosDep,
1494      const vector<const RegressaoLinearSimples> &regressoes)
1495  {
1496      const int linha = ui.tableViewEdicao->currentIndex().row();
1497      const int coluna = ui.tableViewEdicao->currentIndex().column();
1498
1499      const size_t indiceRegistro = _tableViewEdicaoModel->item(linha, 0)->
              text().toLatin1().toInt();
1500
1501      TabelaDeRegistros::obterInstancia()->popularRegistroRLS(indiceRegistro,
               atributosIndep, atributosDep, regressoes);
1502
1503      const Registro &r = TabelaDeRegistros::obterInstancia()->registro(
              indiceRegistro);
1504      vector<string> registroVetor = r.valoresString();
1505      registroVetor.insert(registroVetor.begin(), Utilidades::dblToStdStr(
              indiceRegistro, 0));
1506      Populador::populeLinhaTabela(registroVetor, linha, ui.tableViewEdicao);
1507
1508      atualizeEstatisticaColuna(atributosDep);
1509
1510      ui.tableViewEdicao->setCurrentIndex(_tableViewEdicaoModel->index(linha,
              coluna));
1511  }
1512
1513  void WidgetTabelaEditora::popularColunaRLS(
1514      const vector<string> &atributosIndep,
1515      const vector<string> &atributosDep,
1516      const vector<const RegressaoLinearSimples> &regressoes)
1517  {
1518      const int linha = ui.tableViewEdicao->currentIndex().row();
1519
1520      const size_t indiceRegistro = _tableViewEdicaoModel->item(linha, 0)->
              text().toLatin1().toInt();
1521
1522      TabelaDeRegistros::obterInstancia()->popularColunasRLS(atributosIndep,
              atributosDep, regressoes);
1523
1524      atualizeEstatisticaColuna(atributosDep);
1525
1526      repopularColunas(atributosDep);
1527  }
```

Listagem B.102: widget_tabela_editora.cpp

```
1  #ifndef WIDGET_TABELA_EDITORA_H
2  #define WIDGET_TABELA_EDITORA_H
3
4  #include "ui_widget_tabela_editora.h"
5
6  #include <QModelIndex>
7  #include <QStandardItemModel>
8  #include <QComboBox>
9  #include <QResizeEvent>
10
11 #include "enums_at.h"
12 #include "widget_progresso.h"
13 #include "thread_controller.h"
14 #include "tabela_registros.h"
15 #include "valor_base.h"
16
17 #include <string>
18
19 using namespace std;
20
21 class WidgetTabelaEditora : public WidgetProgresso
22 {
23     Q_OBJECT
24
```

```
25 public:
26     WidgetTabelaEditora(const vector<size_t> &indicesRegistros, QWidget *
           parent = 0);
27     ~WidgetTabelaEditora();
28
29     void destacar(const map<string, vector<size_t>> &indicesPorAtributo);
30     void setBarraMenuVisivel(const bool visivel);
31
32 private:
33     Ui::WidgetTabelaEditora ui;
34     QComboBox *_comboBox;
35     QModelIndex *_indiceCombo;
36     QStandardItemModel *_tableViewEdicaoModel;
37     int _linhaClick;
38     int _colunaClick;
39     size_t _totalLinhas;
40
41     ThreadController _threadController;
42
43     vector<size_t> _indicesTodosRegistros;
44     map<string, vector<size_t>> _registrosDestacadosPorAtributo;
45
46     QMenuBar *_menu;
47
48     void inicializaMenuBarra();
49     void removerComboBox();
50     void removerComboBox(const string &novoValorStr);
51     void habilitarEventos(const bool habilitado) const;
52     bool atributoSelecionado(string &atributo);
53     void focarNosRegistros(const vector<size_t> &registros, const bool
           mostrarTodos) const;
54     void repopularColunas(const vector<string> &atributos);
55     void resetarDestaque();
56     void aposIdentificarRegistro(const vector<size_t> &
           registrosIdentificados) const;
57     void aposRemoverRegistro(const vector<size_t> &registrosRemovidos);
58     void atualizeEstatisticaColuna(const vector<string> &atributos);
59
60 protected:
61 #ifndef TCC_RODOLFO
62     virtual void onUpdateProgresso(StatusThread*) {};
63 #endif
64     virtual void onUpdateProgresso(int *porcentagem);
65     virtual void resizeEvent(QResizeEvent *evento);
66
67 public slots:
68     void removerLinha(const bool removerRegistro = true);
69     void removerLinhasDestacadasDoAtributo(const bool removerRegistro =
           true);
70     void removerTodasLinhas();
71
72     void substituaClick();
73
74 private slots:
75     void fimColetaEstatisticas(const vector<string> *atributos);
76     void itemComboClick(QString textoSelecionado);
77     void mostrarTodos();
78
79     void celulaAlterada(QStandardItem *item);
80     void celulaClick(const QModelIndex &indice);
81     void celulaClickDuplo(const QModelIndex &indice);
82
83     void inserirColunaDialog();
84     void inserirColuna(const string &atributo, const TipoDado_AT tipo);
85     void inserirLinha();
86
87     void renomearColunaDialog();
88     void renomearColuna(const string &nomeAntigo, const string &nomeNovo);
89
90     void removerColuna();
91     void removerLinhas(const vector<size_t> &indiceRegistros, const bool
           removerRegistro = true);
92
93     void removerIdentificarClick();
94     void removerIdentificarQualitativaDialog(const string &
           atributoSelecionado);
95     void removerIdentificarQuantitativoDialog(const string &
           atributoSelecionado);
96     void remIdeOutliersDialog();
97     void removerRegistrosPorValor(const vector<ValorBase *const> &valores,
           const string &atributo, const bool missingsTmb, const bool
           invalidosTmb);
```

```
98      void removerRegistrosQuantPorIntervalo(const vector<TabelaDeRegistros::
            Intervalo> &intervalos, const string &atributo, const bool
            missingsTmb, const bool invalidosTmb);
99
100     void identificarRegistrosPorValor(const vector<ValorBase *const> &
            valores, const string &atributo, const bool missingsTmb, const
            bool invalidosTmb) const;
101     void identificarRegistrosQuantPorIntervalo(const vector<
            TabelaDeRegistros::Intervalo> &intervalos, const string &atributo,
             const bool missingsTmb, const bool invalidosTmb) const;
102
103     void substituaQualitativoDialog(const string &atributoSelecionado);
104     void substituaQuantitativoDialog(const string &atributoSelecionado);
105     void substituaMissings(const string &atributo, ValorBase *const
            novoValor);
106     void substituaTodos(const string &atributo, ValorBase *const novoValor)
            ;
107     void substituaInvalidos(const string &atributo, ValorBase *const
            novoValor);
108     void substituaDestacados(const string &atributo, ValorBase *const
            novoValor);
109     void substituaIntervalos(const string &atributo, ValorBase *const
            novoValor, const vector<TabelaDeRegistros::Intervalo> &intervalos,
             const bool missingsTmb, const bool invalidosTmb);
110     void substituaMissingsDP(const string &atributo, const Distribuicao &
            distribuicao);
111     void substituaTodosDP(const string &atributo, const Distribuicao &
            distribuicao);
112     void substituaInvalidosDP(const string &atributo, const Distribuicao &
            distribuicao);
113     void substituaIntervaloDP(const string &atributo, const Distribuicao &
            distribuicao, const vector<TabelaDeRegistros::Intervalo> &
            intervalos, const bool missingsTmb, const bool invalidosTmb);
114     void substituaDestacadosDP(const string &atributo, const Distribuicao &
            distribuicao);
115     void substituaMissingsRLS(const string &atributoDep, const string &
            atributoInd, const RegressaoLinearSimples &regressao);
116     void substituaTodosRLS(const string &atributoDep, const string &
            atributoInd, const RegressaoLinearSimples &regressao);
117     void substituaInvalidosRLS(const string &atributoDep, const string &
            atributoInd, const RegressaoLinearSimples &regressao);
118     void substituaIntervaloRLS(const string &atributoDep, const string &
            atributoInd, const RegressaoLinearSimples &regressao, const vector
            <TabelaDeRegistros::Intervalo> &intervalos,const bool missingsTmb,
             const bool invalidosTmb);
119     void substituaDestacadosRLS(const string &atributoDep, const string &
            atributoInd, const RegressaoLinearSimples &regressao);
120     void substituaLocalizados(const string &atributo, const vector<
            ValorBase *const> &antigosValores, ValorBase *const novoValor,
            const bool missingsTmb, const bool invalidosTmb);
121
122     void gerarValorDPDialog();
123     void gerarLinhaDPDialog();
124     void gerarColunaDPDialog();
125     void gerarValorRLSDialog();
126     void gerarLinhaRLSDialog();
127     void gerarColunaRLSDialog();
128
129     void popularLinhaDP(const vector<string> &atributos, const vector<const
             Distribuicao> &distribuicoes);
130     void popularColunaDP(const vector<string> &atributos, const vector<
            const Distribuicao> &distribuicoes);
131     void popularLinhaRLS(const vector<string> &atributosIndep, const vector
            <string> &atributosDep, const vector<const RegressaoLinearSimples>
             &regressoes);
132     void popularColunaRLS(const vector<string> &atributosIndep, const
            vector<string> &atributosDep, const vector<const
            RegressaoLinearSimples> &regressoes);
133 };
134
135 #endif // WIDGET_TABELA_EDITORA_H
```

Listagem B.103: widget_tabela_editora.h

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ui version="4.0">
3   <class>WidgetTabelaEditora</class>
4   <widget class="QWidget" name="WidgetTabelaEditora">
5    <property name="geometry">
```

```
    <rect>
     <x>0</x>
     <y>0</y>
     <width>598</width>
     <height>465</height>
    </rect>
   </property>
   <property name="windowTitle">
    <string>WidgetTabelaEditora</string>
   </property>
   <layout class="QVBoxLayout" name="verticalLayout">
    <item>
     <spacer name="verticalSpacer">
      <property name="orientation">
       <enum>Qt::Vertical</enum>
      </property>
      <property name="sizeType">
       <enum>QSizePolicy::Minimum</enum>
      </property>
      <property name="sizeHint" stdset="0">
       <size>
        <width>20</width>
        <height>35</height>
       </size>
      </property>
     </spacer>
    </item>
    <item>
     <widget class="QTableView" name="tableViewEdicao"/>
    </item>
    <item>
     <widget class="QProgressBar" name="progressBar">
      <property name="value">
       <number>24</number>
      </property>
     </widget>
    </item>
   </layout>
  </widget>
  <layoutdefault spacing="6" margin="11"/>
  <resources/>
  <connections/>
 </ui>
```

Listagem B.104: widget_tabela_editora.ui

B.2 SUBPROJETO LIB_ANALISE_TRATAMENTO

```
ï»¿<?xml version="1.0" encoding="utf-8"?>
<Project DefaultTargets="Build" ToolsVersion="4.0" xmlns="http://schemas.
    microsoft.com/developer/msbuild/2003">
  <ItemGroup Label="ProjectConfigurations">
    <ProjectConfiguration Include="Debug|Win32">
      <Configuration>Debug</Configuration>
      <Platform>Win32</Platform>
    </ProjectConfiguration>
    <ProjectConfiguration Include="Release|Win32">
      <Configuration>Release</Configuration>
      <Platform>Win32</Platform>
    </ProjectConfiguration>
  </ItemGroup>
  <ItemGroup>
    <ClInclude Include="distribuicao_controle.h" />
    <ClInclude Include="enums_at.h" />
    <ClInclude Include="gerador_distribuicoes.h" />
    <ClInclude Include="regressao_linear_simples.h" />
    <ClInclude Include="valor_base.h" />
    <ClInclude Include="valor_qualitativo.h" />
    <ClInclude Include="valor_quantitativo.h" />
    <ClInclude Include="coletor_estatisticas.h" />
    <ClInclude Include="gerador_graficos.h" />
    <ClInclude Include="registro.h" />
    <ClInclude Include="tabela_registros.h" />
  </ItemGroup>
```

```
26    <ItemGroup>
27      <ClCompile Include="distribuicao_controle.cpp" />
28      <ClCompile Include="gerador_distribuicoes.cpp" />
29      <ClCompile Include="regressao_linear_simples.cpp" />
30      <ClCompile Include="tabela_registros_get_set_registros.cpp" />
31      <ClCompile Include="tabela_registros_get_valores.cpp" />
32      <ClCompile Include="tabela_registros_identificador.cpp" />
33      <ClCompile Include="tabela_registros_populador.cpp" />
34      <ClCompile Include="tabela_registros_removedor.cpp" />
35      <ClCompile Include="tabela_registros_substituidor.cpp" />
36      <ClCompile Include="valor_qualitativo.cpp" />
37      <ClCompile Include="valor_quantitativo.cpp" />
38      <ClCompile Include="coletor_estatisticas.cpp" />
39      <ClCompile Include="gerador_graficos.cpp" />
40      <ClCompile Include="registro.cpp" />
41      <ClCompile Include="tabela_registros.cpp" />
42    </ItemGroup>
43    <ItemGroup>
44      <ProjectReference Include="..\lib_fit\lib_fit.vcxproj">
45        <Project>{7cc8e7fd-d035-45e8-ad6a-2e1b5dd9c0dc}</Project>
46      </ProjectReference>
47    </ItemGroup>
48    <PropertyGroup Label="Globals">
49      <ProjectGuid>{E16D1D59-22F4-4120-947F-C78AC48714BD}</ProjectGuid>
50      <Keyword>Qt4VSv1.0</Keyword>
51    </PropertyGroup>
52    <Import Project="$(VCTargetsPath)\Microsoft.Cpp.Default.props" />
53    <PropertyGroup Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'"
          Label="Configuration">
54      <ConfigurationType>StaticLibrary</ConfigurationType>
55    </PropertyGroup>
56    <PropertyGroup Condition="'$(Configuration)|$(Platform)'=='Release|Win32'
          " Label="Configuration">
57      <ConfigurationType>StaticLibrary</ConfigurationType>
58    </PropertyGroup>
59    <Import Project="$(VCTargetsPath)\Microsoft.Cpp.props" />
60    <ImportGroup Label="ExtensionSettings">
61    </ImportGroup>
62    <ImportGroup Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'"
          Label="PropertySheets">
63      <Import Project="$(UserRootDir)\Microsoft.Cpp.$(Platform).user.props"
            Condition="exists('$(UserRootDir)\Microsoft.Cpp.$(Platform).user.
            props')" Label="LocalAppDataPlatform" />
64    </ImportGroup>
65    <ImportGroup Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
          Label="PropertySheets">
66      <Import Project="$(UserRootDir)\Microsoft.Cpp.$(Platform).user.props"
            Condition="exists('$(UserRootDir)\Microsoft.Cpp.$(Platform).user.
            props')" Label="LocalAppDataPlatform" />
67    </ImportGroup>
68    <PropertyGroup Label="UserMacros" />
69    <PropertyGroup>
70      <_ProjectFileVersion>10.0.30319.1</_ProjectFileVersion>
71      <CodeAnalysisRuleSet Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">AllRules.ruleset</CodeAnalysisRuleSet>
72      <CodeAnalysisRules Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'" />
73      <CodeAnalysisRuleAssemblies Condition="'$(Configuration)|$(Platform)
            '=='Debug|Win32'" />
74      <CodeAnalysisRuleSet Condition="'$(Configuration)|$(Platform)'=='
            Release|Win32'">AllRules.ruleset</CodeAnalysisRuleSet>
75      <CodeAnalysisRules Condition="'$(Configuration)|$(Platform)'=='Release|
            Win32'" />
76      <CodeAnalysisRuleAssemblies Condition="'$(Configuration)|$(Platform)
            '=='Release|Win32'" />
77      <OutDir Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">$(
            SolutionDir)$(Platform)\$(Configuration)\</OutDir>
78      <OutDir Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">$(
            SolutionDir)$(Platform)\$(Configuration)\</OutDir>
79    </PropertyGroup>
80    <ItemDefinitionGroup Condition="'$(Configuration)|$(Platform)'=='Debug|
          Win32'">
81      <ClCompile>
82        <PreprocessorDefinitions>UNICODE;WIN32;QT_DLL;TCC_RODOLFO;
              LIB_ANALISE_TRATAMENTO_LIB;QT_WIDGETS_LIB;QT_CORE_LIB;%(
              PreprocessorDefinitions)</PreprocessorDefinitions>
83        <AdditionalIncludeDirectories>.;.\GeneratedFiles;.\GeneratedFiles\$(
              ConfigurationName);$(QTDIR)\include;..\lib_utilidades;..\lib_fit
              \includes;$(QTDIR)\include\QtWidgets;$(QTDIR)\include\QtCore;%(
              AdditionalIncludeDirectories)</AdditionalIncludeDirectories>
84        <Optimization>Disabled</Optimization>
85        <DebugInformationFormat>ProgramDatabase</DebugInformationFormat>
```

```
86          <RuntimeLibrary>MultiThreadedDebugDLL</RuntimeLibrary>
87          <TreatWChar_tAsBuiltInType>false</TreatWChar_tAsBuiltInType>
88        </ClCompile>
89        <Link>
90          <SubSystem>Windows</SubSystem>
91          <OutputFile>$(OutDir)\$(ProjectName).dll</OutputFile>
92          <AdditionalLibraryDirectories>$(QTDIR)\lib;$(SolutionDir)$(Platform)\
                $(Configuration)\;%(AdditionalLibraryDirectories)</
                AdditionalLibraryDirectories>
93          <GenerateDebugInformation>true</GenerateDebugInformation>
94          <AdditionalDependencies>qtmaind.lib;Qt5Cored.lib;%(
                AdditionalDependencies);lib_utilidades.lib</
                AdditionalDependencies>
95        </Link>
96        <Lib>
97          <AdditionalDependencies>lib_utilidades.lib;lib_fit.lib;%(
                AdditionalDependencies)</AdditionalDependencies>
98        </Lib>
99        <Lib>
100         <AdditionalLibraryDirectories>$(SolutionDir)$(Platform)\$(
                Configuration);%(AdditionalLibraryDirectories)</
                AdditionalLibraryDirectories>
101       </Lib>
102     </ItemDefinitionGroup>
103     <ItemDefinitionGroup Condition="'$(Configuration)|$(Platform)'=='Release|
            Win32'">
104       <ClCompile>
105         <PreprocessorDefinitions>UNICODE;WIN32;QT_DLL;TCC_RODOLFO;QT_NO_DEBUG
                ;NDEBUG;LIB_ANALISE_TRATAMENTO_LIB;QT_WIDGETS_LIB;QT_CORE_LIB;%(
                PreprocessorDefinitions)</PreprocessorDefinitions>
106         <AdditionalIncludeDirectories>.;..\GeneratedFiles;..\GeneratedFiles\$(
                ConfigurationName);$(QTDIR)\include;..\lib_utilidades;..\lib_fit
                \includes;$(QTDIR)\include\QtWidgets;$(QTDIR)\include\QtCore;%(
                AdditionalIncludeDirectories)</AdditionalIncludeDirectories>
107         <DebugInformationFormat>
108         </DebugInformationFormat>
109         <RuntimeLibrary>MultiThreadedDLL</RuntimeLibrary>
110         <TreatWChar_tAsBuiltInType>false</TreatWChar_tAsBuiltInType>
111       </ClCompile>
112       <Link>
113         <SubSystem>Windows</SubSystem>
114         <OutputFile>$(OutDir)\$(ProjectName).dll</OutputFile>
115         <AdditionalLibraryDirectories>$(QTDIR)\lib;$(SolutionDir)$(Platform)\
                $(Configuration)\;%(AdditionalLibraryDirectories)</
                AdditionalLibraryDirectories>
116         <GenerateDebugInformation>false</GenerateDebugInformation>
117         <AdditionalDependencies>qtmain.lib;Qt5Core.lib;%(
                AdditionalDependencies);lib_utilidades.lib</
                AdditionalDependencies>
118       </Link>
119       <Lib>
120         <AdditionalDependencies>lib_utilidades.lib;lib_fit.lib;%(
                AdditionalDependencies)</AdditionalDependencies>
121       </Lib>
122       <Lib>
123         <AdditionalLibraryDirectories>$(SolutionDir)$(Platform)\$(
                Configuration);%(AdditionalLibraryDirectories)</
                AdditionalLibraryDirectories>
124       </Lib>
125     </ItemDefinitionGroup>
126     <Import Project="$(VCTargetsPath)\Microsoft.Cpp.targets" />
127     <ImportGroup Label="ExtensionTargets">
128     </ImportGroup>
129     <ProjectExtensions>
130       <VisualStudio>
131         <UserProperties UicDir=".\GeneratedFiles" MocDir=".\GeneratedFiles\$(
                ConfigurationName)" MocOptions="" RccDir=".\GeneratedFiles"
                lupdateOnBuild="0" lupdateOptions="" lreleaseOptions=""
                Qt5Version_x0020_Win32="msvc2010_opengl" />
132       </VisualStudio>
133     </ProjectExtensions>
134   </Project>
```

Listagem B.105: lib_analise_tratamento.vcxproj

```
1  ï»¿<?xml version="1.0" encoding="utf-8"?>
2  <Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/
       msbuild/2003">
3    <ItemGroup>
```

```
4       <Filter Include="Source Files">
5         <UniqueIdentifier>{4FC737F1-C7A5-4376-A066-2A32D752A2FF}</
              UniqueIdentifier>
6         <Extensions>cpp;cxx;c;def</Extensions>
7       </Filter>
8       <Filter Include="Header Files">
9         <UniqueIdentifier>{93995380-89BD-4b04-88EB-625FBE52EBFB}</
              UniqueIdentifier>
10        <Extensions>h</Extensions>
11      </Filter>
12      <Filter Include="Form Files">
13        <UniqueIdentifier>{99349809-55BA-4b9d-BF79-8FDBB0286EB3}</
              UniqueIdentifier>
14        <Extensions>ui</Extensions>
15      </Filter>
16      <Filter Include="Resource Files">
17        <UniqueIdentifier>{D9D6E242-F8AF-46E4-B9FD-80ECBC20BA3E}</
              UniqueIdentifier>
18        <Extensions>qrc;*</Extensions>
19        <ParseFiles>false</ParseFiles>
20      </Filter>
21      <Filter Include="Generated Files">
22        <UniqueIdentifier>{71ED8ED8-ACB9-4CE9-BBE1-E00B30144E11}</
              UniqueIdentifier>
23        <Extensions>moc;h;cpp</Extensions>
24        <SourceControlFiles>False</SourceControlFiles>
25      </Filter>
26    </ItemGroup>
27    <ItemGroup>
28      <ClInclude Include="coletor_estatisticas.h">
29        <Filter>Header Files</Filter>
30      </ClInclude>
31      <ClInclude Include="registro.h">
32        <Filter>Header Files</Filter>
33      </ClInclude>
34      <ClInclude Include="tabela_registros.h">
35        <Filter>Header Files</Filter>
36      </ClInclude>
37      <ClInclude Include="gerador_graficos.h">
38        <Filter>Header Files</Filter>
39      </ClInclude>
40      <ClInclude Include="valor_base.h">
41        <Filter>Header Files</Filter>
42      </ClInclude>
43      <ClInclude Include="valor_qualitativo.h">
44        <Filter>Header Files</Filter>
45      </ClInclude>
46      <ClInclude Include="valor_quantitativo.h">
47        <Filter>Header Files</Filter>
48      </ClInclude>
49      <ClInclude Include="gerador_distribuicoes.h">
50        <Filter>Header Files</Filter>
51      </ClInclude>
52      <ClInclude Include="distribuicao_controle.h">
53        <Filter>Header Files</Filter>
54      </ClInclude>
55      <ClInclude Include="regressao_linear_simples.h">
56        <Filter>Header Files</Filter>
57      </ClInclude>
58      <ClInclude Include="enums_at.h">
59        <Filter>Header Files</Filter>
60      </ClInclude>
61    </ItemGroup>
62    <ItemGroup>
63      <ClCompile Include="coletor_estatisticas.cpp">
64        <Filter>Source Files</Filter>
65      </ClCompile>
66      <ClCompile Include="tabela_registros.cpp">
67        <Filter>Source Files</Filter>
68      </ClCompile>
69      <ClCompile Include="gerador_graficos.cpp">
70        <Filter>Source Files</Filter>
71      </ClCompile>
72      <ClCompile Include="registro.cpp">
73        <Filter>Source Files</Filter>
74      </ClCompile>
75      <ClCompile Include="valor_qualitativo.cpp">
76        <Filter>Source Files</Filter>
77      </ClCompile>
78      <ClCompile Include="valor_quantitativo.cpp">
79        <Filter>Source Files</Filter>
80      </ClCompile>
```

```
81      <ClCompile Include="gerador_distribuicoes.cpp">
82        <Filter>Source Files</Filter>
83      </ClCompile>
84      <ClCompile Include="distribuicao_controle.cpp">
85        <Filter>Source Files</Filter>
86      </ClCompile>
87      <ClCompile Include="regressao_linear_simples.cpp">
88        <Filter>Source Files</Filter>
89      </ClCompile>
90      <ClCompile Include="tabela_registros_get_valores.cpp">
91        <Filter>Source Files</Filter>
92      </ClCompile>
93      <ClCompile Include="tabela_registros_get_set_registros.cpp">
94        <Filter>Source Files</Filter>
95      </ClCompile>
96      <ClCompile Include="tabela_registros_populador.cpp">
97        <Filter>Source Files</Filter>
98      </ClCompile>
99      <ClCompile Include="tabela_registros_identificador.cpp">
100       <Filter>Source Files</Filter>
101     </ClCompile>
102     <ClCompile Include="tabela_registros_removedor.cpp">
103       <Filter>Source Files</Filter>
104     </ClCompile>
105     <ClCompile Include="tabela_registros_substituidor.cpp">
106       <Filter>Source Files</Filter>
107     </ClCompile>
108   </ItemGroup>
109 </Project>
```

Listagem B.106: lib_analise_tratamento.vcxproj.filters

```
1  #include "coletor_estatisticas.h"
2  #include "utilidades.h"
3
4  #include <algorithm>
5  #include <cmath>
6  #include <cassert>
7  #include <algorithm>
8  #include <math.h>
9
10 ColetorEstatisticas *ColetorEstatisticas::instancia = NULL;
11
12 const double ColetorEstatisticas::_FATOR_WISKER_SUPERIOR = 1.5;
13 const double ColetorEstatisticas::_FATOR_WISKER_INFEIROR = 1.5;
14 const double ColetorEstatisticas::_ZMIN = -2.5;
15 const double ColetorEstatisticas::_ZMAX = 2.5;
16
17 ColetorEstatisticas *ColetorEstatisticas::obterInstancia()
18 {
19     if (!instancia)
20     {
21         instancia = new ColetorEstatisticas();
22     }
23
24     return instancia;
25 }
26
27 void ColetorEstatisticas::destruirInstancia()
28 {
29     if (instancia)
30     {
31         delete instancia;
32     }
33     instancia = NULL;
34 }
35
36 ColetorEstatisticas::~ColetorEstatisticas()
37 {
38     _estatisticasGerais.clear();
39     _estatisticasQualitativos.clear();
40     _frequenciasQuali.clear();
41     _frequenciasQuantDisc.clear();
42     _frequenciasQuantCont.clear();
43 }
44
45 void ColetorEstatisticas::coletarEstatisticasDosAtributos(
46     const vector<string> &atributos,
47     ThreadWorker *thread)
```

```
48 {
49     const size_t totalAtributos = TabelaDeRegistros::obterInstancia()->
           totalAtributos();
50     size_t i = 0;
51 
52     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
53          citAtributo != atributos.cend();
54          citAtributo++, i++)
55     {
56         coletarEstatisticasDoAtributo(atributos[i]);
57 
58         if (thread)
59         {
60             int *porcentagem = new int(min((int)((i + 1) * 1.0 /
                   totalAtributos * 50 + 50), 99));
61             thread->Notificar(porcentagem);
62         }
63     }
64 }
65 
66 void ColetorEstatisticas::coletarEstatisticasDoAtributo(const string &
       atributo)
67 {
68     int missings = 0;
69     int distintos = 0;
70     int unicos = 0 ;
71 
72     vector<ValorBase *const> valores = vector<ValorBase *const>();
73     TabelaDeRegistros::obterInstancia()->valoresAtributo(atributo, valores)
           ;
74 
75     if (TabelaDeRegistros::obterInstancia()->ehQuant(atributo))
76     {
77         double somaValor = 0; //para calculo media
78         double somaDesvioPadrao = 0; //para calculo desvio padrao
79 
80         double media = numeric_limits<double>::quiet_NaN();
81         double maximo = numeric_limits<double>::quiet_NaN();
82         double whiskerSuperiorDiagramaCaixa = numeric_limits<double>::
               quiet_NaN();
83         double quartilSuperior = numeric_limits<double>::quiet_NaN();
84         double mediana = numeric_limits<double>::quiet_NaN();
85         double quartilInferior = numeric_limits<double>::quiet_NaN();
86         double whiskerInferiorDiagramaCaixa = numeric_limits<double>::
               quiet_NaN();
87         double minimo = numeric_limits<double>::quiet_NaN();
88         vector<double> outliersDiagramaCaixa = vector<double>();
89         vector<double> outliersZScore = vector<double>();
90         double desvioPadrao = numeric_limits<double>::quiet_NaN();
91         double coeficienteVariacao = numeric_limits<double>::quiet_NaN();
92         double amplitude = numeric_limits<double>::quiet_NaN();
93         double desvioInterquartilico = numeric_limits<double>::quiet_NaN();
94         int totalValoresInvalidosOuMissings = 0;
95         int totalValoresValidos = 0;
96 
97         _frequenciasQuantDisc[atributo] = FrequenciasQuantDiscreto();
98         FrequenciasQuantDiscreto &freqQuantValor = _frequenciasQuantDisc[
               atributo];
99 
100        _frequenciasQuantCont[atributo] = FrequenciasQuantContinuo();
101        FrequenciasQuantContinuo &freqQuantClasse = _frequenciasQuantCont[
               atributo];
102 
103        //contando missings e invalidos
104        vector<ValorQuantitativo *const> valoresQuantOrdenados = vector<
               ValorQuantitativo *const>();
105        valoresQuantOrdenados.reserve(valores.size());
106        for (vector<ValorBase *const>::const_iterator citValor = valores.
               cbegin();
107             citValor != valores.cend();
108             citValor++)
109        {
110            ValorQuantitativo *const valor = dynamic_cast<ValorQuantitativo
                   *const>(*citValor);
111 
112            if (valor->missing())
113            {
114                totalValoresInvalidosOuMissings++;
115                missings++;
116                continue;
117            }
118            else if (!valor->formatoValido())
```

```
119                 {
120                     totalValoresInvalidosOuMissings++;
121                     continue;
122                 }
123 
124                 totalValoresValidos++;
125                 valoresQuantOrdenados.push_back(valor);
126             }
127 
128             if (totalValoresValidos > 0)
129             {
130                 minimo = numeric_limits<double>::max(); //min começa sendo uma
                        valor grande;
131                 maximo = -1*numeric_limits<double>::max(); //max começa sendo
                        uma valor pequeno;
132 
133                 //ordenar o vetor, necessario para calculo dos quartis
134                 sort(valoresQuantOrdenados.begin(), valoresQuantOrdenados.end()
                        , comparePointedValueLT<ValorQuantitativo>);
135 
136                 for (vector<ValorQuantitativo *const>::const_iterator citValor
                        = valoresQuantOrdenados.cbegin();
137                     citValor != valoresQuantOrdenados.cend();
138                     citValor++)
139                 {
140                     ValorQuantitativo *const valor = *citValor;
141 
142                     const double valorDbl = valor->valorDbl();
143 
144                     //caso nao tenha ainda a frequencia para valor
145                     if (freqQuantValor.count(valorDbl) == 0)
146                     {
147                         freqQuantValor[valorDbl] = 0;
148                     }
149 
150                     freqQuantValor[valorDbl]++;
151 
152                     //min, max
153                     if (valorDbl < minimo)
154                     {
155                         minimo = valorDbl;
156                     }
157                     if (valorDbl > maximo) //nao pode ser else if, caso tenho
                            só um valor ele é max e min
158                     {
159                         maximo = valorDbl;
160                     }
161 
162                     //para calculo da media
163                     somaValor += valorDbl;
164                 }
165 
166                 distintos = freqQuantValor.size();
167                 unicos = contaUnicosQuant(atributo);
168 
169                 media = somaValor / totalValoresValidos;
170 
171                 quartilSuperior = this->quartilSuperior(valoresQuantOrdenados,
                        totalValoresValidos);
172                 mediana = this->mediana(valoresQuantOrdenados,
                        totalValoresValidos);
173                 quartilInferior = this->quartilInferior(valoresQuantOrdenados,
                        totalValoresValidos);
174                 desvioInterquartilico = quartilSuperior - quartilInferior;
175 
176                 whiskerSuperiorDiagramaCaixa = min(quartilSuperior +
                        _FATOR_WISKER_SUPERIOR * desvioInterquartilico, maximo);
177                 whiskerInferiorDiagramaCaixa = max(quartilInferior -
                        _FATOR_WISKER_INFEIROR * desvioInterquartilico, minimo);
178 
179                 //parte do calculo do desvio padrao
180                 for (vector<ValorQuantitativo *const>::const_iterator citValor
                        = valoresQuantOrdenados.cbegin();
181                     citValor != valoresQuantOrdenados.cend();
182                     citValor++)
183                 {
184                     ValorQuantitativo *const valor = *citValor;
185                     const double valorDbl = valor->valorDbl();
186 
187                     //(x - u)^2
188                     somaDesvioPadrao += (valorDbl - media)*(valorDbl - media);
189                 }
```

```
190
191             desvioPadrao = sqrt(somaDesvioPadrao / totalValoresValidos);
192             coeficienteVariacao = (media != 0) ? desvioPadrao / media :
                    coeficienteVariacao;
193             amplitude = maximo - minimo;
194
195             //distribuição de frequencias por classes
196             double h;
197             int k;
198             if (totalValoresValidos > 200)
199             {
200                 // Scott
201                 h = hScott(totalValoresValidos, desvioPadrao);
202                 k = kDeH(maximo, minimo, h);
203             }
204             else
205             {
206                 //Sturges
207                 k = kSturges(totalValoresValidos);
208                 h = hDeK(maximo, minimo, k);
209             }
210
211             //se tiver poucas classes ou muitas coloca sempre entre 3 e 25
212             if (k < 3 && h != 0)
213             {
214                 k = min(3, totalValoresValidos);
215                 h = hDeK(maximo, minimo, k);
216             }
217             else if (k > 25 && h != 0)
218             {
219                 k = 25;
220                 h = hDeK(maximo, minimo, k);
221             }
222
223             //Construção das classes
224             double x1 = minimo;
225             for (int i = 0; i < k; i++)
226             {
227                 double x2 = x1 + h;
228                 if (i == (k - 1))
229                 {
230                     x2 = maximo;
231                 }
232                 freqQuantClasse[ClasseQuant(x1, x2)] = 0;
233                 x1 = x2;
234             }
235
236             //Contagem das classes e detecao de outliers
237             for (vector<ValorQuantitativo *const>::const_iterator citValor
                    = valoresQuantOrdenados.cbegin();
238                 citValor != valoresQuantOrdenados.cend();
239                 citValor++)
240             {
241                 ValorQuantitativo *const valor = *citValor;
242                 const double valorDbl = valor->valorDbl();
243
244                 FrequenciasQuantContinuo::iterator itClasse =
                        freqQuantClasse.begin();
245                 while (!pertenceAClasse(valorDbl, itClasse->first))
246                 {
247                     itClasse++;
248                 }
249                 (itClasse->second)++;
250
251                 //outliers diagrama de caixa
252                 if (valorDbl > whiskerSuperiorDiagramaCaixa || valorDbl <
                        whiskerInferiorDiagramaCaixa)
253                 {
254                     outliersDiagramaCaixa.push_back(valorDbl);
255                 }
256
257                 //outlires zscore
258                 const double z = (valorDbl - media) / desvioPadrao;
259                 if (z > _ZMAX || z < _ZMIN)
260                 {
261                     outliersZScore.push_back(valorDbl);
262                 }
263             }
264         }
265
266         _estatisticasQualitativos[atributo] = EstatisticaQuant();
267         EstatisticaQuant &estatisticaQuant = _estatisticasQualitativos[
```

```
                atributo];
268             estatisticaQuant.media = media;
269             estatisticaQuant.maximo = maximo;
270             estatisticaQuant.whiskerSuperiorDiagramaCaixa =
                    whiskerSuperiorDiagramaCaixa;
271             estatisticaQuant.quartilSuperior = quartilSuperior;
272             estatisticaQuant.mediana = mediana;
273             estatisticaQuant.quartilInferior = quartilInferior;
274             estatisticaQuant.whiskerInferiorDiagramaCaixa =
                    whiskerInferiorDiagramaCaixa;
275             estatisticaQuant.minimo = minimo;
276             estatisticaQuant.outliersDiagramaCaixa = outliersDiagramaCaixa;
277             estatisticaQuant.outliersZScore = outliersZScore;
278             estatisticaQuant.desvioPadrao = desvioPadrao;
279             estatisticaQuant.coeficienteVariacao = coeficienteVariacao;
280             estatisticaQuant.amplitudeTotal = amplitude;
281             estatisticaQuant.desvioInterquartilico = desvioInterquartilico;
282             estatisticaQuant.totalValoresInvalidosOuMissings =
                    totalValoresInvalidosOuMissings;
283             estatisticaQuant.totalValoresValidos = totalValoresValidos;
284
285         }
286         else //QUALITATIVA
287         {
288             vector<ValorQualitativo *const> valoresQuali = vector<
                    ValorQualitativo *const>();
289             valoresQuali.reserve(valores.size());
290             for (vector<ValorBase *const>::const_iterator citValor = valores.
                    cbegin();
291                 citValor != valores.cend();
292                 citValor++)
293             {
294                 ValorBase *const valor = *citValor;
295
296                 if (valor->missing())
297                 {
298                     missings++;
299                     continue;
300                 }
301
302                 valoresQuali.push_back(dynamic_cast<ValorQualitativo *const>(
                        valor));
303             }
304
305             _frequenciasQuali[atributo] = FrequenciasQuali();
306             FrequenciasQuali &freqQuali = _frequenciasQuali[atributo];
307
308             for (vector<ValorQualitativo *const>::const_iterator citValor =
                    valoresQuali.cbegin();
309                 citValor != valoresQuali.cend();
310                 citValor++)
311             {
312                 ValorQualitativo *const valor = *citValor;
313                 const string &valorStr = valor->valorStr();
314
315                 if (freqQuali.count(valorStr) == 0)
316                 {
317                     freqQuali[valorStr] = 0;
318                 }
319                 freqQuali[valorStr]++;
320             }
321
322             distintos = freqQuali.size();
323             unicos = contaUnicosQuali(atributo);
324         }
325
326         _estatisticasGerais[atributo] = EstatisticaGeral();
327         EstatisticaGeral &estatisticaGeral = _estatisticasGerais[atributo];
328         estatisticaGeral.missings = missings;
329         estatisticaGeral.distintos = distintos;
330         estatisticaGeral.unicos = unicos;
331     }
332
333     double ColetorEstatisticas::quartilInferior(
334         const vector<ValorQuantitativo *const> &valoresQuantOrdenados,
335         const int n) const
336     {
337         if ((n + 3) % 4 == 0)
338         {
339             return valoresQuantOrdenados[((n + 3) / 4.0) - 1]->valorDbl();
340         }
341
```

```
342     const double fator1 = valoresQuantOrdenados[floor((n + 3) / 4.0) - 1]->
            valorDbl();
343     const double fatorInterpolacao = ((n + 3) % 4) / 4.0;
344     const double fator2 = valoresQuantOrdenados[ceil((n + 3) / 4.0) - 1]->
            valorDbl();
345     const double fator3 = valoresQuantOrdenados[floor((n + 3) / 4.0) - 1]->
            valorDbl();
346
347     return fator1 + fatorInterpolacao * (fator2 - fator3);
348 }
349
350 double ColetorEstatisticas::mediana(
351     const vector<ValorQuantitativo *const> &valoresQuantOrdenados,
352     const int n) const
353 {
354     if (n % 2 != 0)
355     {
356         return valoresQuantOrdenados[((n + 1) / 2.0) - 1]->valorDbl();
357     }
358
359     return (valoresQuantOrdenados[(n / 2.0) - 1]->valorDbl() +
            valoresQuantOrdenados[((n + 2) / 2.0) - 1]->valorDbl()) / 2.0;
360 }
361
362 double ColetorEstatisticas::quartilSuperior(
363     const vector<ValorQuantitativo *const> &valoresQuantOrdenados,
364     const int n) const
365 {
366     if (((3 * n) + 1) % 4 == 0)
367     {
368         return valoresQuantOrdenados[(((3 * n) + 1) / 4.0) - 1]->valorDbl()
                ;
369     }
370
371     const double fator1 = valoresQuantOrdenados[floor(((3 * n) + 1) / 4.0)
            - 1]->valorDbl();
372     const double fatorInterpolacao = (((3 * n) + 1) % 4) / 4.0;
373     const double fator2 = valoresQuantOrdenados[ceil(((3 * n) + 1) / 4.0) -
             1]->valorDbl();
374     const double fator3 = valoresQuantOrdenados[floor(((3 * n) + 1) / 4.0)
            - 1]->valorDbl();
375
376     return fator1 + fatorInterpolacao * (fator2 - fator3);
377 }
378
379 int ColetorEstatisticas::contaUnicosQuali(const string &atributo) const
380 {
381     const FrequenciasQuali &frequencias = _frequenciasQuali.at(atributo);
382
383     int unicos = 0;
384     for (FrequenciasQuali::const_iterator citFreq = frequencias.cbegin();
385         citFreq != frequencias.cend();
386         citFreq++)
387     {
388         const pair<string, int> &categoriaFrequencia = *citFreq;
389
390         if (categoriaFrequencia.second == 1)
391         {
392             unicos++;
393         }
394     }
395
396     return unicos;
397 }
398
399 int ColetorEstatisticas::contaUnicosQuant(const string &atributo) const
400 {
401     const FrequenciasQuantDiscreto &frequencias = _frequenciasQuantDisc.at(
            atributo);
402
403     int unicos = 0;
404     for (FrequenciasQuantDiscreto::const_iterator citFreq = frequencias.
            cbegin();
405         citFreq != frequencias.cend();
406         citFreq++)
407     {
408         const pair<double, int> &valorFrequencia = *citFreq;
409
410         if (valorFrequencia.second == 1)
411         {
412             unicos++;
413         }
```

```
414     }
415
416     return unicos;
417 }
418
419 const ColetorEstatisticas::FrequenciasQuali &ColetorEstatisticas::
        frequenciasAtributoQualitativo(const string &atributo) const
420 {
421     return _frequenciasQuali.at(atributo);
422 }
423
424 const ColetorEstatisticas::EstatisticaQuant &ColetorEstatisticas::
        estatisticaAtributoQuantitativo(const string &atributo) const
425 {
426     return _estatisticasQualitativos.at(atributo);
427 }
428
429 const ColetorEstatisticas::EstatisticaGeral &ColetorEstatisticas::
        estatisticaGeralAtributo(const string &atributo) const
430 {
431     return _estatisticasGerais.at(atributo);
432 }
433
434 const ColetorEstatisticas::FrequenciasQuantContinuo &ColetorEstatisticas::
        frequenciasAtributoQuantContinuo(const string &atributo) const
435 {
436     return _frequenciasQuantCont.at(atributo);
437 }
438
439 const ColetorEstatisticas::FrequenciasQuantDiscreto &ColetorEstatisticas::
        frequenciasAtributoQuantDiscreto(const string &atributo) const
440 {
441     return _frequenciasQuantDisc.at(atributo);
442 }
443
444 size_t ColetorEstatisticas::totalCategorias(const string &atributoQuali)
        const
445 {
446     return _frequenciasQuali.at(atributoQuali).size();
447 }
448
449 size_t ColetorEstatisticas::totalClasses(const string &atributoQuant) const
450 {
451     return _frequenciasQuantCont.at(atributoQuant).size();
452 }
453
454 size_t ColetorEstatisticas::indiceCategoriaQual(
455     const string &atributoQualitativo,
456     const string &categoria) const
457 {
458     vector<string> categorias = vector<string>();
459     todasCategoriasAtributoQual(atributoQualitativo, categorias);
460     vector<string>::const_iterator citCategoria = find(categorias.cbegin(),
            categorias.cend(), categoria);
461     if (citCategoria == categorias.cend())
462     {
463         throw exception();
464     }
465
466     return citCategoria - categorias.cbegin();
467 }
468
469 size_t ColetorEstatisticas::indiceClasseQuant(
470     const double valorDbl,
471     const string &atributo) const
472 {
473     const FrequenciasQuantContinuo &frequencias =
            frequenciasAtributoQuantContinuo(atributo);
474     bool pertence = false;
475     size_t indice = -1;
476     int tmp = -1;
477     for (FrequenciasQuantContinuo::const_iterator citFreq = frequencias.
            cbegin();
478          citFreq != frequencias.cend() && !pertence;
479          citFreq++)
480     {
481         const pair<ClasseQuant, int> &classeFrequencia = *citFreq;
482         const ClasseQuant &classe = classeFrequencia.first;
483
484         tmp++;
485         if (pertenceAClasse(valorDbl, classe))
486         {
```

```
487                 pertence = true;
488                 indice = tmp;
489             }
490         }
491 
492         if (indice == -1)
493         {
494             throw exception();
495         }
496 
497         return indice;
498     }
499 
500     void ColetorEstatisticas::removerAtributo(const string &atributo)
501     {
502         map<string, EstatisticaGeral>::const_iterator citGeral =
                _estatisticasGerais.find(atributo);
503         _estatisticasGerais.erase(citGeral);
504 
505         map<string, FrequenciasQuali>::const_iterator citQuali =
                _frequenciasQuali.find(atributo);
506         if (citQuali != _frequenciasQuali.cend())
507         {
508             _frequenciasQuali.erase(citQuali);
509         }
510         else
511         {
512             map<string, EstatisticaQuant>::const_iterator citQuantEst =
                    _estatisticasQualitativos.find(atributo);
513             map<string, FrequenciasQuantDiscreto>::const_iterator citQuant =
                    _frequenciasQuantDisc.find(atributo);
514             map<string, FrequenciasQuantContinuo>::const_iterator
                    citClassesQuant = _frequenciasQuantCont.find(atributo);
515             _estatisticasQualitativos.erase(citQuantEst);
516             _frequenciasQuantDisc.erase(citQuant);
517             _frequenciasQuantCont.erase(citClassesQuant);
518         }
519     }
520 
521     void ColetorEstatisticas::renomerAtributo(const string &atributoAntigo,
            const string &atributoNovo)
522     {
523         map<string, EstatisticaGeral>::const_iterator itEstGeral =
                _estatisticasGerais.find(atributoAntigo);
524         if (itEstGeral == _estatisticasGerais.cend())
525         {
526             throw exception();
527         }
528 
529         _estatisticasGerais[atributoNovo] = itEstGeral->second;
530         _estatisticasGerais.erase(itEstGeral);
531 
532         map<string, FrequenciasQuali>::const_iterator citFreqQuali =
                _frequenciasQuali.find(atributoAntigo);
533         if (citFreqQuali != _frequenciasQuali.cend()) //eh qualitativo
534         {
535             const pair<string, FrequenciasQuali> atributoFrequencias = *
                    citFreqQuali;
536             const FrequenciasQuali &frequencias = atributoFrequencias.second;
537 
538             _frequenciasQuali[atributoNovo] = frequencias;
539             _frequenciasQuali.erase(citFreqQuali);
540         }
541         else //eh quantitativo
542         {
543             map<string, EstatisticaQuant>::const_iterator citEstatisticaQuant =
                        _estatisticasQualitativos.find(atributoAntigo);
544             const pair<string, EstatisticaQuant> &atributoEstatistica = *
                    citEstatisticaQuant;
545             const EstatisticaQuant &estatisticaQuant = atributoEstatistica.
                    second;
546 
547             _estatisticasQualitativos[atributoNovo] = estatisticaQuant;
548             _estatisticasQualitativos.erase(citEstatisticaQuant);
549 
550             map<string, FrequenciasQuantDiscreto>::const_iterator
                    citFreqAtribQuantDisc = _frequenciasQuantDisc.find(
                    atributoAntigo);
551             const pair<string, FrequenciasQuantDiscreto>
                    atributoFrequenciasQuantDisc = *citFreqAtribQuantDisc;
552             const FrequenciasQuantDiscreto &frequenciasQuantDisc =
                    atributoFrequenciasQuantDisc.second;
```

```
553
554             _frequenciasQuantDisc[atributoNovo] = frequenciasQuantDisc;
555             _frequenciasQuantDisc.erase(citFreqAtribQuantDisc);
556
557             map<string, FrequenciasQuantContinuo>::const_iterator
                    citFreqAtribQuantCont = _frequenciasQuantCont.find(
                    atributoAntigo);
558             const pair<string, FrequenciasQuantContinuo>
                    atributoFrequenciasQuantCont = *citFreqAtribQuantCont;
559             const FrequenciasQuantContinuo &frequenciasQuantCont =
                    atributoFrequenciasQuantCont.second;
560
561             _frequenciasQuantCont[atributoNovo] = frequenciasQuantCont;
562             _frequenciasQuantCont.erase(citFreqAtribQuantCont);
563         }
564     }
565
566     void ColetorEstatisticas::todasCategoriasAtributoQual(const string &
            atributo, vector<string> &categoriasRetorno) const
567     {
568         const FrequenciasQuali &frequencias = _frequenciasQuali.at(atributo);
569
570         categoriasRetorno.reserve(frequencias.size());
571
572         for (FrequenciasQuali::const_iterator citFreq = frequencias.cbegin();
573             citFreq != frequencias.cend();
574             citFreq++)
575         {
576             const pair<string, int> atributoFrequencia = *citFreq;
577             const string &categoria = atributoFrequencia.first;
578
579             categoriasRetorno.push_back(categoria);
580         }
581     }
582
583     int ColetorEstatisticas::kDeH(
584         const double max,
585         const double min,
586         const double h) const
587     {
588         int k = ceil((max - min) / h);
589         return k > 1 ? k : 1;
590     }
591
592     double ColetorEstatisticas::hDeK(
593         const double max,
594         const double min,
595         const int k) const
596     {
597         return (max - min) / k;
598     }
599
600     double ColetorEstatisticas::hScott(
601         const int n,
602         const double s) const
603     {
604         double hScott = (3.5 * s) / pow(n, 1/3.0);
605
606         return hScott > 1 ? hScott : 1;
607     }
608
609     int ColetorEstatisticas::kSturges(const int n) const
610     {
611         const double dblN = n;
612
613         //log2(x) = log10(x) / log10(2)
614         return ceil(log10(dblN) / log10(2.0)) + 1; //k = log2(n) + 1
615     }
616
617     bool ColetorEstatisticas::pertenceAClasse(
618         const double valorDbl,
619         const ClasseQuant &classe) const
620     {
621         return  (valorDbl >= classe.first) && (valorDbl <= classe.second);
622     }
623
624     void ColetorEstatisticas::estatisticasStr(
625         const string &atributo,
626         vector<pair<string, double>> &retorno) const
627     {
628         if (TabelaDeRegistros::obterInstancia()->ehQuant(atributo))
629         {
```

```
630         const EstatisticaQuant &estatisticaQuant =
                estatisticaAtributoQuantitativo(atributo);
631         retorno.push_back(pair<string, double>("Média", estatisticaQuant.
                media));
632         retorno.push_back(pair<string, double>("Máximo", estatisticaQuant.
                maximo));
633         retorno.push_back(pair<string, double>("Whisker superior",
                estatisticaQuant.whiskerSuperiorDiagramaCaixa));
634         retorno.push_back(pair<string, double>("Quartil superior",
                estatisticaQuant.quartilSuperior));
635         retorno.push_back(pair<string, double>("Mediana", estatisticaQuant.
                mediana));
636         retorno.push_back(pair<string, double>("Quartil inferior",
                estatisticaQuant.quartilInferior));
637         retorno.push_back(pair<string, double>("Whisker inferior",
                estatisticaQuant.whiskerInferiorDiagramaCaixa));
638         retorno.push_back(pair<string, double>("Mínimo", estatisticaQuant.
                minimo));
639         retorno.push_back(pair<string, double>("Total outliers diagrama de
                caixa", estatisticaQuant.outliersDiagramaCaixa.size()));
640         retorno.push_back(pair<string, double>("Total outliers teste z-
                score", estatisticaQuant.outliersZScore.size()));
641         retorno.push_back(pair<string, double>("Desvio padrão",
                estatisticaQuant.desvioPadrao));
642         retorno.push_back(pair<string, double>("Coeficiente de variação",
                estatisticaQuant.coeficienteVariacao));
643         retorno.push_back(pair<string, double>("Amplitude",
                estatisticaQuant.amplitudeTotal));
644         retorno.push_back(pair<string, double>("Desvio interquartílico",
                estatisticaQuant.desvioInterquartilico));
645         retorno.push_back(pair<string, double>("Valores inválidos ou
                missings", estatisticaQuant.totalValoresInvalidosOuMissings));
646         retorno.push_back(pair<string, double>("Valores válidos",
                estatisticaQuant.totalValoresValidos));
647
648         if (TabelaDeRegistros::obterInstancia()->ehQuantContinuo(atributo))
649         {
650             const FrequenciasQuantContinuo &frequencias =
                    frequenciasAtributoQuantContinuo(atributo);
651
652             for (FrequenciasQuantContinuo::const_iterator citFreq =
                    frequencias.cbegin();
653                  citFreq != frequencias.cend();
654                  citFreq++)
655             {
656                 const pair<ClasseQuant, int> &classeFrequencia = *citFreq;
657                 const ClasseQuant &classe = classeFrequencia.first;
658                 const int frequencia = classeFrequencia.second;
659
660                 const string inferior = Utilidades::dblToStdStr(classe.
                        first);
661
662                 const string superior = Utilidades::dblToStdStr(classe.
                        second);
663
664                 const string concatenacao = inferior + " |-- " + superior;
665
666                 retorno.push_back(pair<string, double>(concatenacao,
                        frequencia));
667             }
668         }
669         else if (TabelaDeRegistros::obterInstancia()->ehQuantDiscreto(
                atributo))
670         {
671             const FrequenciasQuantDiscreto &frequencias =
                    frequenciasAtributoQuantDiscreto(atributo);
672
673             for (FrequenciasQuantDiscreto::const_iterator citFreq =
                    frequencias.cbegin();
674                  citFreq != frequencias.cend();
675                  citFreq++)
676             {
677                 const pair<double, int> &valorFrequencia = *citFreq;
678                 const double valor = valorFrequencia.first;
679                 const int frequencia = valorFrequencia.second;
680
681                 retorno.push_back(pair<string, double>(Utilidades::
                        dblToStdStr(valor, 3), frequencia));
682             }
683         }
684     }
685     else
```

```
686     {
687         const FrequenciasQuali &frequencias =
                frequenciasAtributoQualitativo(atributo);
688
689         for (FrequenciasQuali::const_iterator citFreq = frequencias.cbegin
                ();
690              citFreq != frequencias.cend();
691              citFreq++)
692         {
693             const pair<string, int> categoriaFrequencia = *citFreq;
694             const string &categoria = categoriaFrequencia.first;
695             const int frequencia = categoriaFrequencia.second;
696
697             retorno.push_back(pair<string, double>(categoria, frequencia));
698         }
699     }
700 }
701
702 void ColetorEstatisticas::outliersDiagramaCaixa(
703     const string &atributo,
704     const double fatorWS,
705     const double fatorWI,
706     map<size_t, ValorQuantitativo *const> &outliersRetorno) const
707 {
708     const EstatisticaQuant &estatistica = _estatisticasQualitativos.at(
            atributo);
709
710     const TabelaDeRegistros::MapRegistros &registrosMap = TabelaDeRegistros
            ::obterInstancia()->registros();
711     const size_t indiceAtributo = TabelaDeRegistros::obterInstancia()->
            indiceAtributo(atributo);
712
713     const double desvioInterquartilico = estatistica.desvioInterquartilico;
714     const double quartilSuperior = estatistica.quartilSuperior;
715     const double quartilInferior = estatistica.quartilInferior;
716
717     const double whiskerS = quartilSuperior + fatorWS *
            desvioInterquartilico;
718     const double whiskerI = quartilInferior - fatorWI *
            desvioInterquartilico;
719
720     for (TabelaDeRegistros::MapRegistros::const_iterator citRegistro =
            registrosMap.cbegin();
721          citRegistro != registrosMap.cend();
722          citRegistro++)
723     {
724         const size_t indiceRegistro = citRegistro->first;
725         const Registro &registro = citRegistro->second;
726
727         ValorQuantitativo *const valor = registro.retorneValorQuant(
                indiceAtributo);
728         const double valorDbl = valor->valorDbl();
729
730         if ((valorDbl > whiskerS || valorDbl < whiskerI) && outliersRetorno
                .find(indiceRegistro) == outliersRetorno.cend())
731         {
732             outliersRetorno.insert(pair<size_t, ValorQuantitativo *const>(
                    indiceRegistro, valor));
733         }
734     }
735 }
736
737 void ColetorEstatisticas::outliersZScore(
738     const string &atributo,
739     const double zMin,
740     const double zMax,
741     map<size_t, ValorQuantitativo *const> &outliersRetorno) const
742 {
743     const EstatisticaQuant &estatistica = _estatisticasQualitativos.at(
            atributo);
744
745     const TabelaDeRegistros::MapRegistros &registrosMap = TabelaDeRegistros
            ::obterInstancia()->registros();
746     const size_t indiceAtributo = TabelaDeRegistros::obterInstancia()->
            indiceAtributo(atributo);
747
748     const double media = estatistica.media;
749     const double desvioPadrao= estatistica.desvioPadrao;
750
751     for (TabelaDeRegistros::MapRegistros::const_iterator citRegistro =
            registrosMap.cbegin();
752          citRegistro != registrosMap.cend();
```

```
753         citRegistro++)
754     {
755         const size_t indiceRegistro = citRegistro->first;
756         const Registro &registro = citRegistro->second;
757 
758         ValorQuantitativo *const valor = registro.retorneValorQuant(
                indiceAtributo);
759         const double valorDbl = valor->valorDbl();
760 
761         const double z = (valorDbl - media) / desvioPadrao;
762         if ((z > zMax || z < zMin) && outliersRetorno.find(indiceRegistro)
                == outliersRetorno.cend())
763         {
764             outliersRetorno.insert(pair<size_t, ValorQuantitativo *const>(
                    indiceRegistro, valor));
765         }
766     }
767 }
768 
769 double ColetorEstatisticas::calcMediaParcialBaseadoValores(
770     const string &atributo,
771     const string &atributoFiltro,
772     const vector<ValorBase *const> &valoresFiltro) const
773 {
774     vector<size_t> registrosIdentificados = vector<size_t>();
775     TabelaDeRegistros::obterInstancia()->identificarRegistrosPorValor(
            atributoFiltro, valoresFiltro, false, false,
            registrosIdentificados);
776 
777     return calcMediaParcial(atributo, registrosIdentificados);
778 }
779 
780 double ColetorEstatisticas::calcMediaParcialBaseadoIntervalos(
781         const string &atributo,
782         const string &atributoFiltro,
783         const vector<TabelaDeRegistros::Intervalo> &intervalos) const
784 {
785     vector<size_t> registrosIdentificados = vector<size_t>();
786     TabelaDeRegistros::obterInstancia()->
            identificarRegistrosQuantPorIntervalo(atributoFiltro, intervalos,
            false, false, registrosIdentificados);
787 
788     return calcMediaParcial(atributo, registrosIdentificados);
789 }
790 
791 double ColetorEstatisticas::calcMediaParcial(
792     const string &atributo,
793     const vector<size_t> &indices) const
794 {
795     vector<ValorQuantitativo *const> valores = vector<ValorQuantitativo *
            const>();
796     TabelaDeRegistros::obterInstancia()->valoresAtributoQuantPorIndice(
            atributo, indices, valores);
797 
798     double soma = 0;
799     double i = 0;
800 
801     for (vector<ValorQuantitativo *const>::const_iterator citValor =
            valores.cbegin();
802         citValor != valores.cend();
803         citValor++)
804     {
805         ValorQuantitativo *const valor = *citValor;
806 
807         if (!valor->missing() && valor->formatoValido())
808         {
809             soma += valor->valorDbl();
810             i++;
811         }
812     }
813 
814     if (i != 0)
815     {
816         return soma / i;
817     }
818 
819     return numeric_limits<double>::quiet_NaN();
820 }
```

Listagem B.107: coletor_estatisticas.cpp

```
1  #ifndef COLETOR_ESTATISTICAS_H
2  #define COLETOR_ESTATISTICAS_H
3
4  #include "tabela_registros.h"
5
6  using namespace std;
7
8  class ColetorEstatisticas
9  {
10 public:
11     typedef pair<double, double> ClasseQuant;
12     typedef map<ClasseQuant, int> FrequenciasQuantContinuo;
13     typedef map<double, int> FrequenciasQuantDiscreto;
14     typedef map<string, int> FrequenciasQuali;
15
16     static const double _FATOR_WISKER_SUPERIOR;
17     static const double _FATOR_WISKER_INFEIROR;
18     static const double _ZMIN;
19     static const double _ZMAX;
20
21     struct EstatisticaGeral
22     {
23         int missings;
24         int distintos;
25         int unicos;
26     };
27
28     struct EstatisticaQuant
29     {
30         double media;
31         double maximo;
32         double whiskerSuperiorDiagramaCaixa;
33         double quartilSuperior;
34         double mediana;
35         double quartilInferior;
36         double whiskerInferiorDiagramaCaixa;
37         double minimo;
38         double desvioPadrao;
39         double coeficienteVariacao;
40         double amplitudeTotal;
41         double desvioInterquartilico;
42
43         vector<double> outliersDiagramaCaixa;
44         vector<double> outliersZScore;
45
46         int totalValoresInvalidosOuMissings;
47         int totalValoresValidos;
48     };
49
50     static ColetorEstatisticas *obterInstancia();
51     static void destruirInstancia();
52
53     double quartilSuperior(const vector<ValorQuantitativo *const> &
           valoresQuantOrdenados, const int n) const;
54     double mediana(const vector<ValorQuantitativo *const> &
           valoresQuantOrdenados, const int n) const;
55     double quartilInferior(const vector<ValorQuantitativo *const> &
           valoresQuantOrdenados, const int n) const;
56
57     const EstatisticaGeral &estatisticaGeralAtributo(const string &atributo
           ) const;
58     const EstatisticaQuant &estatisticaAtributoQuantitativo(const string &
           atributo) const;
59     const FrequenciasQuali &frequenciasAtributoQualitativo(const string &
           atributo) const;
60     const FrequenciasQuantContinuo &frequenciasAtributoQuantContinuo(const
           string &atributo) const;
61     const FrequenciasQuantDiscreto &frequenciasAtributoQuantDiscreto(const
           string &atributo) const;
62     size_t totalCategorias(const string &atributoQuali) const;
63     size_t totalClasses(const string &atributoQuant) const;
64     void estatisticasStr(const string &atributo, vector<pair<string, double
           >> &retorno) const;
65     void todasCategoriasAtributoQual(const string &atributo, vector<string>
            &valoresRetorno) const;
66
67     void removerAtributo(const string &atributo);
68     void renomerAtributo(const string &atributo, const string &nomeNovo);
69
70     size_t indiceCategoriaQual(const string &atributo, const string &
           categoria) const;
```

```
71      size_t indiceClasseQuant(const double valorDbl, const string &atributo)
            const;
72
73      void coletarEstatisticasDosAtributos(const vector<string> &atributos,
            ThreadWorker *thread = NULL);
74
75      double calcMediaParcialBaseadoValores(const string &atributo, const
            string &atributoFiltro, const vector<ValorBase *const> &
            valoresFiltro) const;
76      double calcMediaParcialBaseadoIntervalos(const string &atributo, const
            string &atributoFiltro, const vector<TabelaDeRegistros::Intervalo>
             &intervalos) const;
77      void outliersDiagramaCaixa(const string &atributo, const double fatorWS
            , const double fatorWI, map<size_t, ValorQuantitativo *const> &
            outliersRetorno) const;
78      void outliersZScore(const string &atributo, const double zMin, const
            double zMax, map<size_t, ValorQuantitativo *const> &
            outliersRetorno) const;
79
80  private:
81      static ColetorEstatisticas *instancia;
82
83      map<string, EstatisticaGeral> _estatisticasGerais;
84      map<string, EstatisticaQuant> _estatisticasQualitativos;
85      map<string, FrequenciasQuali> _frequenciasQuali;
86      map<string, FrequenciasQuantDiscreto> _frequenciasQuantDisc;
87      map<string, FrequenciasQuantContinuo> _frequenciasQuantCont;
88
89      ColetorEstatisticas(){};
90      ColetorEstatisticas(const ColetorEstatisticas&){};
91      ColetorEstatisticas &operator =(const ColetorEstatisticas&){};
92
93      ~ColetorEstatisticas();
94
95      bool pertenceAClasse(const double valorDbl, const ClasseQuant &classe)
            const;
96      double calcMediaParcial(const string &atributo, const vector<size_t> &
            indices) const;
97      int contaUnicosQuali(const string &atributo) const;
98      int contaUnicosQuant(const string &atributo) const;
99      void coletarEstatisticasDoAtributo(const string &atributo);
100
101     double hDeK(const double max, const double min, const int k) const;
102     double hScott(const int n, const double s) const;
103     int kDeH(const double max, const double min, const double h) const;
104     int kSturges(const int n) const;
105 };
106 #endif
```

Listagem B.108: coletor_estatisticas.h

```
1   #include "distribuicao_controle.h"
2
3   Distribuicao::Distribuicao(const fit::_distribution_type tipo) :
4   _fitDistribuicao(fit::get_default_distribution(tipo)->clone()),
5       _tipo(tipo),
6       _valorP(0),
7       _valorPValido(false)
8   {}
9
10  Distribuicao::Distribuicao(const list<double> &valores, const fit::
        _distribution_type tipo) :
11  _fitDistribuicao(NULL),
12      _tipo(tipo),
13      _valorP(0),
14      _valorPValido(false)
15  {
16      gerarDistribuicaoDeValores(valores);
17  }
18
19  Distribuicao::Distribuicao(const vector<double> &parametros,
20      const list<double> &valores,
21      const fit::_distribution_type tipo) :
22  _fitDistribuicao(NULL),
23      _tipo(tipo),
24      _valorP(0),
25      _valorPValido(false)
26  {
27      gerarDistribuicaoDeParametros(parametros, valores);
```

```
28 }
29
30 Distribuicao::Distribuicao(const Distribuicao &outra) :
31     _fitDistribuicao((outra._fitDistribuicao != NULL) ? outra.
           _fitDistribuicao->clone() : NULL),
32     _tipo(outra._tipo),
33     _valorP(outra._valorP),
34     _valorPValido(outra._valorPValido)
35 {}
36
37 Distribuicao &Distribuicao::operator= (const Distribuicao &outra)
38 {
39     if (_fitDistribuicao)
40     {
41         delete _fitDistribuicao;
42     }
43
44     _fitDistribuicao = (outra._fitDistribuicao != NULL) ? outra.
           _fitDistribuicao->clone() : NULL;
45     _tipo = outra._tipo;
46     _valorP = outra._valorP;
47     _valorPValido = outra._valorPValido;
48
49     return *this;
50 }
51
52 Distribuicao::~Distribuicao()
53 {
54     if (_fitDistribuicao)
55     {
56         delete _fitDistribuicao;
57         _fitDistribuicao = NULL;
58     }
59 }
60
61 void Distribuicao::gerarDistribuicaoDeValores(const list<double> &valores)
62 {
63     if (_fitDistribuicao)
64     {
65         delete _fitDistribuicao;
66         _fitDistribuicao = NULL;
67     }
68
69     _fitDistribuicao = fit::get_distribution(valores, _tipo)->clone();
70
71     calcularValorP(valores);
72 }
73
74 void Distribuicao::gerarDistribuicaoDeParametros(
75     const vector<double> &parametros,
76     const list<double> &valores)
77 {
78     if (_fitDistribuicao)
79     {
80         delete _fitDistribuicao;
81         _fitDistribuicao = NULL;
82     }
83
84     _fitDistribuicao = fit::get_distribution(parametros, _tipo)->clone();
85
86     calcularValorP(valores);
87 }
88
89 void Distribuicao::calcularValorP(const list<double> &valores)
90 {
91     if (valores.empty())
92     {
93         _valorP = 0;
94         _valorPValido = false;
95         return;
96     }
97
98     if (!_fitDistribuicao)
99     {
100        _fitDistribuicao = fit::get_distribution(valores, _tipo)->clone();
101    }
102
103    fit::freq_table freqTable = fit::get_freq_table(valores, *(
           _fitDistribuicao), fit::STURGES);
104    _valorP = fit::get_chisquare_test(freqTable).chi_p_value;
105    _valorPValido = true;
106 }
```

```
107
108 bool Distribuicao::operator <(const Distribuicao &outra) const
109 {
110     return this->_valorP < outra._valorP;
111 }
112
113 double Distribuicao::valorP() const
114 {
115     return _valorP;
116 }
117
118 bool Distribuicao::valorPValido() const
119 {
120     return _valorPValido;
121 }
122
123 fit::_distribution_type Distribuicao::tipo() const
124 {
125     return _tipo;
126 }
127
128 string Distribuicao::nome() const
129 {
130     return _fitDistribuicao->name();
131 }
132 vector<string> Distribuicao::nomesParametros() const
133 {
134     return _fitDistribuicao->paramNames();
135 }
136
137 vector<double> Distribuicao::parametros() const
138 {
139     return _fitDistribuicao->get_params();
140 }
141
142 double Distribuicao::gerarValorAleatorio() const
143 {
144     return _fitDistribuicao->get_random();
145 }
```

Listagem B.109: distribuicao_controle.cpp

```
1  #ifndef DISTRIBUICAO_H
2  #define DISTRIBUICAO_H
3
4  #ifndef TCC_RODOLFO
5  #include "lib_fit/includes/fit.h"
6  #else
7  #include "../lib_fit/includes/fit.h"
8  #endif
9
10 using namespace std;
11
12 class Distribuicao
13 {
14 public:
15     Distribuicao(const Distribuicao &outra);
16     Distribuicao(const fit::_distribution_type tipo);
17     Distribuicao(const list<double> &valores, const fit::_distribution_type
           tipo);
18     Distribuicao(const vector<double> &parametros, const list<double> &
           valores, const fit::_distribution_type tipo);
19     ~Distribuicao();
20
21     Distribuicao &operator= (const Distribuicao &outra);
22     bool operator <(const Distribuicao &outra) const;
23
24     bool valorPValido() const;
25     double gerarValorAleatorio() const;
26     double valorP() const;
27     fit::_distribution_type tipo() const;
28     string nome() const;
29     vector<double> parametros() const;
30     vector<string> nomesParametros() const;
31
32 private:
33     fit::base_distrib *_fitDistribuicao;
34     fit::_distribution_type _tipo;
35     double _valorP;
```

```
36      bool _valorPValido;
37
38      void gerarDistribuicaoDeValores(const list<double> &valores);
39      void gerarDistribuicaoDeParametros(const vector<double> &parametros,
            const list<double> &valores);
40      void calcularValorP(const list<double> &valores);
41  };
42
43  #endif
```

Listagem B.110: distribuicao_controle.h

```
1  #ifndef ENUMS_AT_H
2  #define ENUMS_AT_H
3
4  enum TipoDado_AT {QUALITATIVO = 0, QUANT_DISCRETO = 1, QUANT_CONTINUO = 2};
5  enum Operacao {MAIORQUE_MENORQUE = 0, MAIORQUE_MENORIGUALQUE = 1,
       MAIORIGUALQUE_MENORQUE = 2, MAIORIGUALQUE_MENORIGUALQUE = 3};
6
7  #endif
```

Listagem B.111: enums_at.h

```
1  #include "gerador_distribuicoes.h"
2
3  #include <iterator>
4  #include <utility>
5  #include <algorithm>
6
7  #include "utilidades.h"
8
9  const fit::_distribution_type GeradorDistribuicoes::tiposDistri[8] = {fit::
       BETA, fit::EXPONENTIAL, fit::GAMMA, fit::LOGNORMAL, fit::NORMAL, fit::
       TRIANGULAR, fit::UNIFORM, fit::WEIBULL};
10
11 void GeradorDistribuicoes::gerarTodasDistribuicoes(
12     const list<double> &valores,
13     vector<const Distribuicao> &distrsOkRetorno,
14     vector<const Distribuicao> &distrsNaoOkRetorno)
15 {
16     bool erro = true;
17
18     for (size_t i = 0; i < 8; i++)
19     {
20         if (!valores.empty())
21         {
22             try
23             {
24                 distrsOkRetorno.push_back(Distribuicao(valores, tiposDistri
                       [i]));
25                 erro = false;
26             }
27             catch(...)
28             {
29                 erro = true;
30             }
31         }
32
33         if (erro)
34         {
35             try
36             {
37                 distrsNaoOkRetorno.push_back(Distribuicao(tiposDistri[i]));
38             }
39             catch(...)
40             {
41             }
42         }
43     }
44
45     sort(distrsOkRetorno.begin(), distrsOkRetorno.end());
46 }
```

Listagem B.112: gerador_distribuicoes.cpp

```
#ifndef MANIPULADORDISTRIBUICOES_H
#define MANIPULADORDISTRIBUICOES_H

#include "distribuicao_controle.h"

using namespace std;

class GeradorDistribuicoes
{
public:
    static void gerarTodasDistribuicoes(const list<double> &valores, vector
        <const Distribuicao> &distrsOkRetorno, vector<const Distribuicao>
        &distrsNaoOkRetorno);
private:
    static const fit::_distribution_type tiposDistri[8];
};
#endif
```

Listagem B.113: gerador_distribuicoes.h

```
#include "gerador_graficos.h"

#include "utilidades.h"

void GeradorGraficos::gerarHistogramaQuantContinuo(
    const string &atributo,
    vector<ColetorEstatisticas::ClasseQuant> &classesRetorno,
    Frequencias &frequenciasRetorno)
{
    const ColetorEstatisticas::FrequenciasQuantContinuo &freq =
        ColetorEstatisticas::obterInstancia()->
        frequenciasAtributoQuantContinuo(atributo);

    const size_t totalClasses = freq.size();

    for (ColetorEstatisticas::FrequenciasQuantContinuo::const_iterator
        citFreq = freq.cbegin();
        citFreq != freq.cend();
        citFreq++)
    {
        const pair<ColetorEstatisticas::ClasseQuant, int> &classeFrequencia
            = *citFreq;
        const ColetorEstatisticas::ClasseQuant &classe = classeFrequencia.
            first;
        const int frequencia = classeFrequencia.second;

        frequenciasRetorno.push_back(frequencia);
        classesRetorno.push_back(classe);
    }
}

void GeradorGraficos::gerarGraficoBarrasQuantDiscreto(
    const string &atributo,
    vector<string> &categoriasRetorno,
    Frequencias &frequenciasRetorno)
{
    const ColetorEstatisticas::FrequenciasQuantDiscreto &frequencias =
        ColetorEstatisticas::obterInstancia()->
        frequenciasAtributoQuantDiscreto(atributo);

    for (ColetorEstatisticas::FrequenciasQuantDiscreto::const_iterator
        citFreq = frequencias.cbegin();
        citFreq != frequencias.cend();
        citFreq++)
    {
        const pair<double, int> &valorFrequencia = *citFreq;
        const double valor = valorFrequencia.first;
        const int frequencia = valorFrequencia.second;

        frequenciasRetorno.push_back(frequencia);
        categoriasRetorno.push_back(Utilidades::dblToStdStr(valor));
    }
}

void GeradorGraficos::gerarGraficoBarrasQualitativo(
    const string &atributo,
    vector<string> &categoriasRetorno,
    Frequencias &frequenciasRetorno)
{
```

```
52      const ColetorEstatisticas::FrequenciasQuali &frequencias =
            ColetorEstatisticas::obterInstancia()->
            frequenciasAtributoQualitativo(atributo);
53
54      for (ColetorEstatisticas::FrequenciasQuali::const_iterator citFreq =
            frequencias.cbegin();
55          citFreq != frequencias.cend();
56          citFreq++)
57      {
58          const pair<string, int> categoriaFrequencia = *citFreq;
59          const string &categoria = categoriaFrequencia.first;
60          const int frequencia = categoriaFrequencia.second;
61
62          frequenciasRetorno.push_back(frequencia);
63          categoriasRetorno.push_back(categoria);
64      }
65  }
66
67  void GeradorGraficos::gerarDiagramaDispersao2v(
68      const string &atributoX,
69      const string &atributoY,
70      Coordenadas &coordenadasRetorno)
71  {
72      const size_t indiceAtribX = TabelaDeRegistros::obterInstancia()->
            indiceAtributo(atributoX);
73      const size_t indiceAtribY = TabelaDeRegistros::obterInstancia()->
            indiceAtributo(atributoY);
74
75      const TabelaDeRegistros::MapRegistros &regs = TabelaDeRegistros::
            obterInstancia()->registros();
76
77      for (TabelaDeRegistros::MapRegistros::const_iterator citRegistro = regs
            .cbegin();
78          citRegistro != regs.cend();
79          citRegistro++)
80      {
81          const Registro &registro= citRegistro->second;
82
83          ValorBase *const valorX = registro.retorneValor(indiceAtribX);
84          ValorBase *const valorY = registro.retorneValor(indiceAtribY);
85
86          if (!valorX->missing() && !valorY->missing())
87          {
88              double x = posicaoNoEixo(atributoX, valorX);
89              double y = posicaoNoEixo(atributoY, valorY);
90
91              Coordenada c;
92              c.push_back(x);
93              c.push_back(y);
94
95              coordenadasRetorno.push_back(c);
96          }
97      }
98  }
99
100 void GeradorGraficos::gerarDiagramaDispersao3v(
101     const string &atributoX,
102     const string &atributoY,
103     const string &atributoZ,
104     Coordenadas &coordenadasRetorno)
105 {
106     const size_t indiceAtribX = TabelaDeRegistros::obterInstancia()->
            indiceAtributo(atributoX);;
107     const size_t indiceAtribY = TabelaDeRegistros::obterInstancia()->
            indiceAtributo(atributoY);;
108     const size_t indiceAtribZ = TabelaDeRegistros::obterInstancia()->
            indiceAtributo(atributoZ);;
109
110     // registros
111     const TabelaDeRegistros::MapRegistros &regs = TabelaDeRegistros::
            obterInstancia()->registros();
112
113     for (TabelaDeRegistros::MapRegistros::const_iterator citRegistro = regs
            .cbegin();
114         citRegistro != regs.cend();
115         citRegistro++)
116     {
117         const Registro &registro= citRegistro->second;
118
119         ValorBase *const valorX = registro.retorneValor(indiceAtribX);
120         ValorBase *const valorY = registro.retorneValor(indiceAtribY);
121         ValorBase *const valorZ = registro.retorneValor(indiceAtribZ);
```

```
122
123             if (!valorX->missing() && !valorY->missing() && !valorZ->missing())
124             {
125                 double x = posicaoNoEixo(atributoX, valorX);
126                 double y = posicaoNoEixo(atributoY, valorY);
127                 double z = posicaoNoEixo(atributoZ, valorZ);
128
129                 if (isnan(x) || isnan(y) || isnan(z)) {continue;} //not a
                        number
130
131                 Coordenada c;
132                 c.push_back(x);
133                 c.push_back(y);
134                 c.push_back(z);
135
136                 coordenadasRetorno.push_back(c);
137             }
138         }
139 }
140
141 void GeradorGraficos::gerarGraficoMultBarras(
142         const string &atributoX,
143         const string &atributoY,
144         vector<Frequencias> &frequenciasRetorno)
145 {
146         size_t n = 0;
147         size_t m = 0;
148
149         if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributoX))
150         {
151             n = ColetorEstatisticas::obterInstancia()->totalCategorias(
                    atributoX);
152         }
153         else
154         {
155             n = ColetorEstatisticas::obterInstancia()->totalClasses(atributoX);
156         }
157
158         if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributoY))
159         {
160             m = ColetorEstatisticas::obterInstancia()->totalCategorias(
                    atributoY);
161         }
162         else
163         {
164             m = ColetorEstatisticas::obterInstancia()->totalClasses(atributoY);
165         }
166
167         // frequências inicias iguais a zero.
168         for (size_t i = 0; i < n; i++)
169         {
170             Frequencias fequencia = Frequencias();
171             for (size_t j = 0; j < m; j++)
172             {
173                 fequencia.push_back(0);
174             }
175
176             frequenciasRetorno.push_back(fequencia);
177         }
178
179         const size_t indiceAtribX = TabelaDeRegistros::obterInstancia()->
                indiceAtributo(atributoX);
180         const size_t indiceAtribY = TabelaDeRegistros::obterInstancia()->
                indiceAtributo(atributoY);
181
182         // registros
183         const TabelaDeRegistros::MapRegistros &regs = TabelaDeRegistros::
                obterInstancia()->registros();
184
185         for (TabelaDeRegistros::MapRegistros::const_iterator citRegistro = regs
                .cbegin();
186             citRegistro != regs.cend();
187             citRegistro++)
188         {
189             const Registro &registro= citRegistro->second;
190
191             ValorBase *const valorX = registro.retorneValor(indiceAtribX);
192             ValorBase *const valorY = registro.retorneValor(indiceAtribY);
193
194             if (!valorX->missing() && !valorY->missing())
195             {
196                 size_t linha = -1;
```

```
197             if (TabelaDeRegistros::obterInstancia()->ehQualitativo(
                    atributoX))
198             {
199                 linha = ColetorEstatisticas::obterInstancia()->
                        indiceCategoriaQual(atributoX, valorX->valorStr());
200             }
201             else
202             {
203                 const double x = dynamic_cast<ValorQuantitativo *const>(
                        valorX)->valorDbl();
204                 if (isnan(x)) {continue;} //if not a number
205                 linha = ColetorEstatisticas::obterInstancia()->
                        indiceClasseQuant(x, atributoX);
206             }
207
208             size_t coluna = -1;
209             if (TabelaDeRegistros::obterInstancia()->ehQualitativo(
                    atributoY))
210             {
211                 coluna = ColetorEstatisticas::obterInstancia()->
                        indiceCategoriaQual(atributoY, valorY->valorStr());
212             }
213             else
214             {
215                 const double y = dynamic_cast<ValorQuantitativo *const>(
                        valorY)->valorDbl();
216                 if (isnan(y)) {continue;} //if not a number
217                 coluna = ColetorEstatisticas::obterInstancia()->
                        indiceClasseQuant(y, atributoY);
218             }
219             if (linha != -1 && coluna != -1)
220             {
221                 frequenciasRetorno[linha][coluna]++;
222             }
223         }
224     }
225 }
226
227 double GeradorGraficos::posicaoNoEixo(
228     const string &atributo,
229     ValorBase *const valor)
230 {
231     double posicaoNoEixo = 0.0;
232     if (TabelaDeRegistros::obterInstancia()->ehQualitativo(atributo))
233     {
234         posicaoNoEixo = ColetorEstatisticas::obterInstancia()->
                indiceCategoriaQual(atributo, valor->valorStr());
235     }
236     else
237     {
238         posicaoNoEixo = dynamic_cast<ValorQuantitativo *const>(valor)->
                valorDbl();
239     }
240     return posicaoNoEixo;
241 }
```

Listagem B.114: gerador_graficos.cpp

```
1  #ifndef GERADORGRAFICOS_H
2  #define GERADORGRAFICOS_H
3
4  #include "coletor_estatisticas.h"
5
6  using namespace std;
7
8  class GeradorGraficos
9  {
10 public:
11     typedef vector<double> Coordenada;
12     typedef vector<Coordenada> Coordenadas;
13     typedef vector<int> Frequencias;
14
15     static void gerarDiagramaDispersao2v(const string &atributoX, const
           string &atributoY, Coordenadas &coordenadasRetorno);
16     static void gerarDiagramaDispersao3v(const string &atributoX, const
           string &atributoY, const string &atributoZ, Coordenadas &
           coordenadasRetorno);
17     static void gerarGraficoBarrasQualitativo(const string &atributo,
           vector<string> &categoriasRetorno, Frequencias &frequenciasRetorno
           );
```

```
18      static void gerarHistogramaQuantContinuo(const string &atributo, vector
            <ColetorEstatisticas::ClasseQuant> &classesRetorno, Frequencias &
            frequenciasRetorno);
19      static void gerarGraficoBarrasQuantDiscreto(const string &atributo,
            vector<string> &categoriasRetorno, Frequencias &frequenciasRetorno
            );
20      static void gerarGraficoMultBarras(const string &atributoX, const
            string &atributoY, vector<Frequencias> &retorno);
21
22  private:
23      static double posicaoNoEixo(const string &atributo, ValorBase *const
            valor);
24  };
25  #endif
```

Listagem B.115: gerador_graficos.h

```
1   #include "registro.h"
2
3   Registro::Registro()
4   {
5       _valores = vector<ValorBase *const>();
6   }
7
8   Registro::Registro(const Registro &outro)
9   {
10      _valores.clear();
11
12      for (vector<ValorBase *const>::const_iterator citValorOutro = outro.
            _valores.cbegin();
13          citValorOutro != outro._valores.cend();
14          citValorOutro++)
15      {
16          ValorBase *const valorOutro = *citValorOutro;
17          this->_valores.push_back(valorOutro->clone());
18      }
19  }
20  Registro &Registro::operator= (const Registro &outro)
21  {
22      for (vector<ValorBase *const>::const_iterator citValor = this->_valores
            .cbegin();
23          citValor != this->_valores.cend();
24          citValor++)
25      {
26          ValorBase *const valor = *citValor;
27          delete valor;
28      }
29
30      _valores.clear();
31
32      for (vector<ValorBase *const>::const_iterator citValorOutro = outro.
            _valores.cbegin();
33          citValorOutro != outro._valores.cend();
34          citValorOutro++)
35      {
36          ValorBase *const valorOutro = *citValorOutro;
37          this->_valores.push_back(valorOutro->clone());
38      }
39
40      return *this;
41  }
42
43  Registro::~Registro()
44  {
45      for (vector<ValorBase *const>::const_iterator citValor = _valores.
            cbegin();
46          citValor != _valores.cend();
47          citValor++)
48      {
49          ValorBase *const valor = *citValor;
50          delete valor;
51      }
52
53      _valores.clear();
54  }
55
56  void Registro::insiraValor(ValorBase *const valor)
57  {
58      _valores.push_back(valor->clone());
```

```
59 }
60
61 void Registro::removaValor(const size_t indice)
62 {
63     delete _valores[indice];
64     _valores.erase(_valores.cbegin() + indice);
65 }
66
67 void Registro::substituaValor(const size_t indice, ValorBase *const
       novoValor)
68 {
69     delete _valores[indice];
70     _valores[indice] = novoValor->clone();
71 }
72
73 ValorBase *const Registro::retorneValor(const size_t indice) const
74 {
75     return _valores[indice];
76 }
77
78 ValorQuantitativo *const Registro::retorneValorQuant(const size_t indice)
       const
79 {
80     return dynamic_cast<ValorQuantitativo *const>(_valores[indice]);
81 }
82 ValorQualitativo *const Registro::retorneValorQual(const size_t indice)
       const
83 {
84     return dynamic_cast<ValorQualitativo *const>(_valores[indice]);
85 }
86
87 bool Registro::algumValorInvalidoOuMissingNosIndices(const vector<size_t>
       indices) const
88 {
89     for (vector<size_t>::const_iterator citIndice = indices.cbegin();
90          citIndice != indices.cend();
91          citIndice++)
92     {
93         const size_t indice = *citIndice;
94         if (indice > _valores.size())
95         {
96             return true;
97         }
98
99         ValorBase *const valorBase = _valores[indice];
100        ValorQuantitativo *const valorQuant = dynamic_cast<
               ValorQuantitativo *const>(valorBase);
101
102        if (valorBase->missing() || (valorQuant && !valorQuant->
               formatoValido()))
103        {
104            return true;
105        }
106    }
107
108    return false;
109 }
110
111 bool Registro::algumValorInvalidoOuMissing() const
112 {
113     for (vector<ValorBase *const>::const_iterator citValor = _valores.
           cbegin();
114          citValor != _valores.cend();
115          citValor++)
116     {
117         ValorBase *const valorBase = *citValor;
118         ValorQuantitativo *const valorQuant = dynamic_cast<
               ValorQuantitativo *const>(valorBase);
119
120         if (valorBase->missing() || (valorQuant && !valorQuant->
               formatoValido()))
121         {
122             return true;
123         }
124     }
125
126     return false;
127 }
128
129 const vector<ValorBase *const> &Registro::valores() const
130 {
131     return _valores;
```

```
132 }
133
134 vector<string> Registro::valoresString() const
135 {
136     vector<string> valoresString = vector<string>();
137     for (vector<ValorBase *const>::const_iterator citValor = _valores.
          cbegin();
138          citValor != _valores.cend();
139          citValor++)
140     {
141         ValorBase *const valor = *citValor;
142         valoresString.push_back(valor->valorStr());
143     }
144
145     return valoresString;
146 }
```

Listagem B.116: registro.cpp

```
1  #ifndef REGISTRO_H
2  #define REGISTRO_H
3
4  #include "valor_qualitativo.h"
5  #include "valor_quantitativo.h"
6
7  #include <vector>
8
9  using namespace std;
10
11 class Registro
12 {
13 public:
14     Registro();
15     Registro(const Registro &outro);
16     ~Registro();
17     Registro &operator= (const Registro &outro);
18     void insiraValor(ValorBase *const valor);
19     void removaValor(const size_t indice);
20     void substituaValor(const size_t indice, ValorBase *const novoValor);
21     ValorBase *const retorneValor(const size_t indice) const;
22     ValorQuantitativo *const retorneValorQuant(const size_t indice) const;
23     ValorQualitativo *const retorneValorQual(const size_t indice) const;
24     bool algumValorInvalidoOuMissingNosIndices(const vector<size_t> indices
           ) const;
25     bool algumValorInvalidoOuMissing() const;
26     const vector<ValorBase *const> &valores() const;
27     vector<string> valoresString() const;
28 private:
29     vector<ValorBase *const> _valores;
30 };
31 #endif
```

Listagem B.117: registro.h

```
1  #include "regressao_linear_simples.h"
2
3  RegressaoLinearSimples::RegressaoLinearSimples(const vector<double> &
       valoresIndependentes, const vector<double> &valoresDependentes)
4  {
5      const size_t n = min(valoresIndependentes.size(), valoresDependentes.
           size());
6
7      //E = somatorio
8      double Ex = 0;
9      double Ex2 = 0;
10     double Ey = 0;
11     double Exy = 0;
12
13     for (size_t i = 0; i < n; i++)
14     {
15         const double x = valoresIndependentes.at(i);
16         const double y = valoresDependentes.at(i);
17
18         Ex += x;
19         Ex2 += x * x;
20         Ey += y;
```

```
21          Exy += x * y;
22      }
23
24      _beta = ((n * Exy) - (Ex * Ey)) / ((n * Ex2) - pow(Ex, 2));
25
26      const double mediaX = Ex / n;
27      const double mediaY = Ey / n;
28
29      _alpha = mediaY - _beta * mediaX;
30  }
31
32  RegressaoLinearSimples::~RegressaoLinearSimples()
33  {
34
35  }
36
37  double RegressaoLinearSimples::alpha() const
38  {
39      return _alpha;
40  }
41
42  double RegressaoLinearSimples::beta() const
43  {
44      return _beta;
45  }
46
47  double RegressaoLinearSimples::gerarValorDependente(const double
        valorIndependente) const
48  {
49      return _alpha + _beta * valorIndependente;
50  }
```

Listagem B.118: regressao_linear_simples.cpp

```
1  #ifndef REGRESSA_LINEAR_SIMPLES_H
2  #define REGRESSA_LINEAR_SIMPLES_H
3
4  #include <vector>
5
6  using namespace std;
7
8  class RegressaoLinearSimples
9  {
10 public:
11     RegressaoLinearSimples(){};
12     RegressaoLinearSimples(const vector<double> &valoresIndependentes,
           const vector<double> &valoresDependentes);
13     ~RegressaoLinearSimples();
14     double gerarValorDependente(const double valorIndependente) const;
15     double alpha() const;
16     double beta() const;
17 private:
18     double _alpha;
19     double _beta;
20 };
21
22 #endif
```

Listagem B.119: regressao_linear_simples.h

```
1  #include "tabela_registros.h"
2
3  #include "utilidades.h"
4
5  #include <algorithm>
6  #include <cassert>
7  #include <set>
8
9  #include <QCoreApplication>
10
11 TabelaDeRegistros *TabelaDeRegistros::instancia = NULL;
12
13 TabelaDeRegistros *TabelaDeRegistros::obterInstancia()
14 {
15     if (!instancia)
16     {
```

```
17          instancia = new TabelaDeRegistros();
18      }
19
20      return instancia;
21  }
22
23  void TabelaDeRegistros::destruirInstancia()
24  {
25      if (instancia)
26      {
27          delete instancia;
28      }
29      instancia = NULL;
30  }
31
32  TabelaDeRegistros::TabelaDeRegistros()
33  {
34      _totalRegistros = 0;
35      _novoRegistroIndice = 0;
36  }
37
38  TabelaDeRegistros::~TabelaDeRegistros()
39  {
40      _registros.clear();
41      _atributos.clear();
42      _tipos.clear();
43  }
44
45  void TabelaDeRegistros::inicialize(
46      const vector<vector<ValorBase *const>> &dadosEmColuna,
47      const vector<string> &atributos,
48      const vector<TipoDado_AT> &tipos,
49      ThreadWorker *thread)
50  {
51      assert(!dadosEmColuna.empty());
52      assert(!dadosEmColuna.front().empty());
53      assert(!atributos.empty());
54      assert(!tipos.empty());
55
56      _atributos.clear();
57      for (vector<string>::const_iterator citAtributo = atributos.cbegin();
58          citAtributo != atributos.cend();
59          citAtributo++)
60      {
61          string atributo = *citAtributo;
62
63          int i = 2;
64          while (std::find(_atributos.cbegin(), _atributos.cend(), atributo)
                    != _atributos.cend())
65          {
66              atributo += "_" +Utilidades::dblToStdStr(i, 0);
67          }
68
69          _atributos.push_back(atributo);
70      }
71
72      _tipos.clear();
73      for (vector<TipoDado_AT>::const_iterator citTipo = tipos.cbegin();
74          citTipo != tipos.cend();
75          citTipo++)
76      {
77          const TipoDado_AT tipo = *citTipo;
78          _tipos.push_back(tipo);
79      }
80
81      _totalRegistros = dadosEmColuna[0].size();
82
83      const size_t totalAtributos = dadosEmColuna.size();
84
85      const int taxaNotificacao = _totalRegistros <= 10 ? 10 :
             _totalRegistros / 10;
86
87      _registros.clear();
88
89      size_t i = 0;
90      for (i = 0; i < _totalRegistros; i++)
91      {
92          _registros[i] = Registro();
93          Registro &r = _registros[i];
94
95          if ((i % taxaNotificacao) == 0 &&  thread)
96          {
```

```
97              int *porcentagem = new int(min((int)((i + 1) * 1.0 /
                    _totalRegistros * 50), 99));
98              thread->Notificar(porcentagem);
99          }
100
101         for (size_t j = 0; j < totalAtributos; j++)
102         {
103             r.insiraValor(dadosEmColuna.at(j).at(i));
104             delete dadosEmColuna.at(j).at(i);
105         }
106     }
107
108     _novoRegistroIndice = i;
109 }
110
111 void TabelaDeRegistros::adicionarAtributo(
112     const string &novoAtributo,
113     const TipoDado_AT tipo)
114 {
115     _atributos.push_back(novoAtributo);
116     _tipos.push_back(tipo);
117
118     ValorBase *valor = TabelaDeRegistros::gerarValor(novoAtributo, "");
119
120     for (MapRegistros::iterator itRegistro = _registros.begin();
121         itRegistro != _registros.end();
122         itRegistro++)
123     {
124         Registro &registro = itRegistro->second;
125         registro.insiraValor(valor);
126     }
127
128     delete valor;
129 }
130
131 TipoDado_AT TabelaDeRegistros::tipo(const string &atributo) const
132 {
133     return _tipos[this->indiceAtributo(atributo)];
134 }
135
136 void TabelaDeRegistros::todosIndicesRegistros(vector<size_t> &
        indicesRetorno) const
137 {
138     indicesRetorno.reserve(_totalRegistros);
139
140     for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
141         citRegistro != _registros.cend();
142         citRegistro++)
143     {
144         const size_t indiceRegistro = citRegistro->first;
145         indicesRetorno.push_back(indiceRegistro);
146     }
147 }
148
149 void TabelaDeRegistros::removerAtributo(const string &atributo)
150 {
151     const size_t indiceAtributo = this->indiceAtributo(atributo);
152
153     // 1 - Remove da lista de nomes e de tipos.
154     _atributos.erase(_atributos.cbegin() + indiceAtributo);
155     _tipos.erase(_tipos.cbegin() + indiceAtributo);
156
157     // 2 - Remove dos registros da tabela.
158     for (MapRegistros::iterator itRegistro = _registros.begin();
159         itRegistro != _registros.end();
160         itRegistro++)
161     {
162         Registro &registro = itRegistro->second;
163         registro.removaValor(indiceAtributo);
164     }
165 }
166
167 bool TabelaDeRegistros::pertenceAosIntervalos(
168     const vector<TabelaDeRegistros::Intervalo> &intervalos,
169     const double valorDbl) const
170 {
171     Operacao op;
172     double inferior;
173     double superior;
174
175     for (vector<TabelaDeRegistros::Intervalo>::const_iterator intervalo =
            intervalos.cbegin();
```

```
176             intervalo != intervalos.cend();
177             intervalo++)
178         {
179             inferior = std::get<0>(*intervalo);
180             superior = std::get<1>(*intervalo);
181             op = std::get<2>(*intervalo);
182 
183             switch (op)
184             {
185             case MAIORQUE_MENORQUE:
186                 {
187                     if (valorDbl > inferior && valorDbl < superior)
188                     {
189                         return true;
190                     }
191                     break;
192                 }
193             case MAIORQUE_MENORIGUALQUE:
194                 {
195                     if (valorDbl > inferior && valorDbl <= superior)
196                     {
197                         return true;
198                     }
199                     break;
200                 }
201             case MAIORIGUALQUE_MENORQUE:
202                 {
203                     if (valorDbl >= inferior && valorDbl < superior)
204                     {
205                         return true;
206                     }
207                     break;
208                 }
209             case MAIORIGUALQUE_MENORIGUALQUE:
210                 {
211                     if (valorDbl >= inferior && valorDbl <= superior)
212                     {
213                         return true;
214                     }
215                     break;
216                 }
217             }
218         }
219         return false;
220 }
221 
222 void TabelaDeRegistros::renomerAtributo(
223         const string &atributo,
224         const string &novoNome)
225 {
226         _atributos[this->indiceAtributo(atributo)] = novoNome;
227 }
228 
229 void TabelaDeRegistros::substituaTodosPorIndice(const string &atributo,
             ValorBase *const valorNovo, const vector<size_t> &quaisIndices)
230 {
231         const size_t indiceAtributo = this->indiceAtributo(atributo);
232 
233         for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
234             citIndice != quaisIndices.cend();
235             citIndice++)
236         {
237             const size_t indice = *citIndice;
238             Registro &registro = _registros.at(indice);
239 
240             registro.substituaValor(indiceAtributo, valorNovo);
241         }
242 }
243 
244 const vector<string> &TabelaDeRegistros::atributos() const
245 {
246         return _atributos;
247 }
248 
249 vector<string> TabelaDeRegistros::atributosQuantitativos() const
250 {
251         vector<string> retorno = vector<string>();
252         for (vector<string>::const_iterator citAtributo = _atributos.cbegin();
253             citAtributo != _atributos.cend();
254             citAtributo++)
255         {
256             const string &atributo = *citAtributo;
```

```
257
258         if (ehQuant(atributo))
259         {
260             retorno.push_back(atributo);
261         }
262     }
263
264     return retorno;
265 }
266
267 vector<string> TabelaDeRegistros::atributosQualitativos() const
268 {
269     vector<string> retorno = vector<string>();
270     for (vector<string>::const_iterator citAtributo = _atributos.cbegin();
271         citAtributo != _atributos.cend();
272         citAtributo++)
273     {
274         const string &atributo = *citAtributo;
275
276         if (ehQualitativo(atributo))
277         {
278             retorno.push_back(atributo);
279         }
280     }
281
282     return retorno;
283 }
284
285 const TabelaDeRegistros::MapRegistros &TabelaDeRegistros::registros() const
286 {
287     return _registros;
288 }
289
290 const vector<TipoDado_AT> &TabelaDeRegistros::tipos() const
291 {
292     return _tipos;
293 }
294
295 size_t TabelaDeRegistros::totalAtributos() const
296 {
297     return _atributos.size();
298 }
299
300 size_t TabelaDeRegistros::totalRegistros() const
301 {
302     return _totalRegistros;
303 }
304
305 size_t TabelaDeRegistros::indiceAtributo(const string &atributo) const
306 {
307     vector<string>::const_iterator cit = find(_atributos.cbegin(),
          _atributos.cend(), atributo);
308
309     if (cit != _atributos.cend())
310     {
311         return cit - _atributos.cbegin();
312     }
313
314     return -1;
315 }
316
317 vector<size_t> TabelaDeRegistros::indicesAtributos(const vector<string> &
      atributos) const
318 {
319     vector<size_t> indices = vector<size_t>();
320
321     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
322         citAtributo != atributos.cend();
323         citAtributo++)
324     {
325         const string &atributo = *citAtributo;
326
327         vector<string>::const_iterator cit = find(_atributos.cbegin(),
              _atributos.cend(), atributo);
328
329         if (cit != _atributos.cend())
330         {
331             indices.push_back(cit - _atributos.cbegin());
332         }
333     }
334
335     return indices;
```

```
336 }
337
338 bool TabelaDeRegistros::ehQuant(const string &atributo) const
339 {
340     const size_t i = this->indiceAtributo(atributo);
341     return _tipos[i] == QUANT_CONTINUO || _tipos[i] == QUANT_DISCRETO;
342 }
343
344 bool TabelaDeRegistros::ehQuantContinuo(const string &atributo) const
345 {
346     const size_t i = this->indiceAtributo(atributo);
347     return _tipos[i] == QUANT_CONTINUO;
348 }
349
350 bool TabelaDeRegistros::ehQuantDiscreto(const string &atributo) const
351 {
352     const size_t i = this->indiceAtributo(atributo);
353     return _tipos[i] == QUANT_DISCRETO;
354 }
355
356 bool TabelaDeRegistros::ehVariavelReferencia(const string &atributo, vector
      <int> &valoresFaltantesRetorno, string &msg) const
357 {
358     msg = "";
359
360     vector<ValorQuantitativo *const> valores = vector<ValorQuantitativo *
          const>();
361     set<int> valoresInt = set<int>();
362     valoresAtributoQuant(atributo, true, valores);
363
364     int max = std::numeric_limits<int>::min();
365     int min = std::numeric_limits<int>::max();
366
367     for (vector<ValorQuantitativo *const>::const_iterator citValor =
           valores.cbegin();
368          citValor != valores.cend();
369          citValor++)
370     {
371         ValorQuantitativo *const valor = *citValor;
372
373         if (valor->missing() || !valor->formatoValido())
374         {
375             msg = "O atributo de referência não pode conter valores
                    missings/inválidos!";
376             return false;
377         }
378
379         const double valorDbl = valor->valorDbl();
380         if (!Utilidades::is_integer(valorDbl))
381         {
382             msg = "O atributo de referência não pode conter valores
                    fracionários!";
383             return false;
384         }
385
386         if (valoresInt.find(valorDbl) != valoresInt.end())
387         {
388             msg = "O atributo de referência não pode conter valores
                    repetidos!";
389             return false;
390         }
391
392         if (valorDbl > max)
393         {
394             max = valorDbl;
395         }
396
397         if (valorDbl < min)
398         {
399             min = valorDbl;
400         }
401
402         valoresInt.insert(valorDbl);
403     }
404
405     set<int> todos = set<int>();
406
407     for (int valor = min; valor <= max; valor++)
408     {
409         todos.insert(valor);
410     }
411
```

```
412     for (set<int>::const_iterator citInt = valoresInt.cbegin();
413          citInt != valoresInt.cend();
414          citInt++)
415     {
416         const int valorInt = *citInt;
417
418         set<int>::iterator it = todos.find(valorInt);
419         if (it != todos.cend())
420         {
421             todos.erase(it);
422         }
423     }
424
425     copy(todos.begin(),todos.end(), back_inserter(valoresFaltantesRetorno))
            ;
426     return true;
427 }
428
429 bool TabelaDeRegistros::ehQualitativo(const string &atributo) const
430 {
431     const size_t i = this->indiceAtributo(atributo);
432     return _tipos[i] == QUALITATIVO;
433 }
434
435 bool TabelaDeRegistros::existeAtributo(const string &atributo) const
436 {
437     return std::find(_atributos.cbegin(), _atributos.cend(), atributo) !=
            _atributos.cend();
438 }
439
440 void TabelaDeRegistros::dadosStringToValoresBases(
441     vector<vector<string>> &stringColunas,
442     const vector<TipoDado_AT> &tipos,
443     vector<vector<ValorBase *const>> &valoresColunasRetorno,
444     QProgressDialog *progress)
445 {
446     const size_t totalColunas = tipos.size();
447     const size_t totalLinhas = stringColunas[0].size();
448
449     for (size_t coluna = 0; coluna < totalColunas; coluna++)
450     {
451         valoresColunasRetorno.push_back(vector<ValorBase *const>());
452
453         for (size_t linha = 0; linha < totalLinhas; linha++)
454         {
455             valoresColunasRetorno.back().push_back(gerarValor(tipos[coluna
                    ], stringColunas[coluna][linha]));
456         }
457
458         if (progress)
459         {
460             int a = (coluna + 1) / totalColunas * 100;
461
462             progress->setValue(min((int)((coluna + 1) * 1.0 / totalColunas
                        * 100), 99));
463             QCoreApplication::processEvents(QEventLoop::
                    ExcludeUserInputEvents);
464         }
465     }
466 }
467
468 ValorBase *TabelaDeRegistros::gerarValor(
469     const TipoDado_AT tipo,
470     const string &valor)
471 {
472     if (tipo == QUANT_CONTINUO || tipo == QUANT_DISCRETO)
473     {
474         return new ValorQuantitativo(valor);
475     }
476
477     return new ValorQualitativo(valor);
478 }
479
480 ValorBase *TabelaDeRegistros::gerarValor(
481     const string &atributo,
482     const string &valor) const
483 {
484     return gerarValor(_tipos.at(this->indiceAtributo(atributo)), valor);
485 }
486
487 ValorBase *TabelaDeRegistros::gerarValorQuant(const double valorDbl)
488 {
```

```
489     return new ValorQuantitativo(valorDbl);
490 }
```

Listagem B.120: tabela_registros.cpp

```
 1 #ifndef TABELADEREGISTROS_H
 2 #define TABELADEREGISTROS_H
 3
 4 #include <map>
 5 #include <tuple>
 6 #include "thread_worker.h"
 7 #include "registro.h"
 8 #include "enums_at.h"
 9 #include "distribuicao_controle.h"
10 #include "regressao_linear_simples.h"
11
12 #include <QProgressDialog>
13
14 using namespace std;
15
16 class TabelaDeRegistros
17 {
18 public:
19     typedef map<size_t, Registro> MapRegistros;
20     typedef tuple<double, double, Operacao> Intervalo;
21
22     static TabelaDeRegistros *obterInstancia();
23     static void destruirInstancia();
24
25     bool ehQualitativo(const string &atributo) const;
26     bool ehQuant(const string &atributo) const;
27     bool ehQuantContinuo(const string &atributo) const;
28     bool ehQuantDiscreto(const string &atributo) const;
29     bool ehVariavelReferencia(const string &atributo, vector<int> &
           valoresFaltantes, string &msg) const;
30     bool existeAtributo(const string &atributo) const;
31     const MapRegistros &registros() const;
32     const vector<string> &atributos() const;
33     const vector<TipoDado_AT> &tipos() const;
34     size_t totalRegistros() const;
35     size_t indiceAtributo(const string &atributo) const;
36     size_t totalAtributos() const;
37     static void dadosStringToValoresBases(vector<vector<string>> &
           stringColunas, const vector<TipoDado_AT> &tipos, vector<vector<
           ValorBase *const>> &valoresColunasRetorno, QProgressDialog *
           progress = NULL);
38     TipoDado_AT tipo(const string &atributo) const;
39     ValorBase *gerarValor(const string &atributo, const string &valor)
           const;
40     static ValorBase *gerarValor(const TipoDado_AT tipo, const string &
           valor);
41     static ValorBase *gerarValorQuant(const double valorDbl);
42     vector<size_t> indicesAtributos(const vector<string> &atributos) const;
43     vector<string> atributosQualitativos() const;
44     vector<string> atributosQuantitativos() const;
45     void adicionarAtributo(const string &novoAtributo, const TipoDado_AT
           tipo);
46     void removerAtributo(const string &atributo);
47     void renomerAtributo(const string &atributo, const string &novoNome);
48     void inicialize(const vector<vector<ValorBase *const>> &dadosEmColuna,
           const vector<string> &atributos, const vector<TipoDado_AT> &tipos,
            ThreadWorker *thread = NULL);
49     void todosIndicesRegistros(vector<size_t> &indicesRetorno) const;
50
51     //tabela_registros_get_set_registros.cpp ------
52     const Registro &registro(const size_t indiceRegistro) const;
53     size_t novoRegistro();
54     void atualizeRegistro(const size_t indiceRegistro, const string &
           atributo, ValorBase *const novoValor);
55     void indiceRegistrosComplementares(vector<size_t> &indicesRegistros,
           vector<size_t> &indicesComplementaresRetorno) const;
56     void indiceRegistrosComplementares(vector<vector<size_t>> &
           indicesRegistrosColunas, vector<vector<size_t>> &
           indicesComplementaresColunasRetorno) const;
57     void registrosPorIndice(const vector<size_t> quaisIndices, vector<const
            Registro *const> &registrosRetorno) const;
58     void registrosVetorStrPorIndice(const vector<size_t> quaisIndices,
           vector<vector<string>> &registrosRetorno, const bool comId = false
           ) const;
```

```
59      void todosRegistros(vector<const Registro *const> &registrosRetorno)
            const;
60      void todosRegistrosVetorStr(vector<vector<string>> &registrosRetorno,
            const bool comId = false) const;
61      //-------------------------------------------------------
62
63      //tabela_registro_populador.cpp ------------
64      void popularColunasDP(const vector<string> &atributos, const vector<
            const Distribuicao> &distribuicoes);
65      void popularColunasRLS(const vector<string> &atributosIndep, const
            vector<string> &atributosDep, const vector<const
            RegressaoLinearSimples> &regressoes);
66      void popularRegistroDP(const size_t indiceRegistro, const vector<string
            > &atributos, const vector<const Distribuicao> &distribuicoes);
67      void popularRegistroRLS(const size_t indiceRegistro, const vector<
            string> &atributosIndep, const vector<string> &atributosDep, const
             vector<const RegressaoLinearSimples> &regressoes);
68      //-------------------------------------------------------
69
70      //tabela_registros_get_valores.cpp ---------
71      void valoresAtributo(const string &atributo, vector<ValorBase *const> &
            valoresRetorno) const;
72      void valoresAtributoQuant(const string &atributo, const bool
            todosMissingsInvalidos, vector<ValorQuantitativo *const> &
            valoresRetorno) const;
73      void valoresAtributoQuantPorIndice(const string &atributo, const vector
            <size_t> &quaisIndices, vector<ValorQuantitativo *const> &
            valoresRetorno) const;
74      void valoresAtributoQuantDbl(const string &atributo, const bool
            todosMissingsInvalidos, vector<double> &valoresRetorno);
75      void valoresAtributosQuantDbl(const vector<string> &atributos, const
            bool todosMissingsInvalidos, vector<vector<double>> &
            valoresRetorno) const;
76      void valoresAtributosQuantDblPorIndice(const vector<string> &atributos,
             const vector<size_t> &quaisIndices, const bool
            todosMissingsInvalidos, vector<vector<double>> &valoresRetorno)
            const;
77      void valoresAtributosQuantDblPorIndicePorAtributo(const vector<string>
            &atributos, const vector<vector<size_t>> &quaisIndicesPorAtributo,
             vector<vector<double>> &valoresRetorno) const;
78      void valoresAtributoStr(const string &atributo, map<size_t, string> &
            valoresRetorno) const;
79      void valoresAtributoStrPorIndice(const string &atributo, const vector<
            size_t> &quaisIndices, map<size_t, string> &valoresRetorno) const;
80      //-------------------------------------------------------
81
82      //tabela_registros_identificador.cpp ---------
83      void identificarRegistrosMissingsInvalidos(vector<size_t> &
            registrosMissInvaRetorno, map<string, vector<size_t>> &
            missInvPorAtribRetorno) const;
84      void identificarRegistrosPorValor(const string &atributo, const vector<
            ValorBase *const> &valores, const bool missingsTmb, bool
            invalidosTmb, vector<size_t> &indicesIdentificadosRetorno) const;
85      void identificarRegistrosQuantPorIntervalo(const string &atributo,
            const vector<TabelaDeRegistros::Intervalo> &intervalos, const bool
             missingsTmb, const bool invalidosTmb, vector<size_t> &
            indicesIdentificadosRetorno) const;
86      void identificarRegistrosQuantPorIntervaloPorIndice(const string &
            atributo, const vector<size_t> &quaisIndices, const vector<
            TabelaDeRegistros::Intervalo> &intervalos, const bool missingsTmb,
             const bool invalidosTmb, vector<size_t> &
            indicesIdentificadosRetorno) const;
87      void identificarRegistrosMultiAtributos(const vector<string> &
            atribsQuant, const vector<vector<Intervalo>> &intervalos, const
            vector<bool> &missingsQuantTmb, const vector<bool> &
            invalidosQuantTmb, const vector<string> &atribsQual, const vector<
            vector<ValorBase *const>> &valores, const vector<bool> &
            missingsQualTmb, const vector<bool> &invalidosQualTmb, vector<
            size_t> &indicesIdentificadosRetorno) const;
88      //-------------------------------------------------------
89
90      //tabela_registros_removedor.cpp -----------------
91      void removaRegistro(const size_t indiceRegistro);
92      void removaRegistrosPorValores(const string &atributo, const vector<
            ValorBase *const> &valores, const bool missingsTmb, const bool
            invalidosTmb, vector<size_t> &indicesRemovidosRetorno);
93      void removerRegistrosQuantPorIntervalo(const vector<TabelaDeRegistros::
            Intervalo> &intervalos, const string &atributo, const bool
            missingsTmb, const bool invalidosTmb, vector<size_t> &
            indicesRemovidosRetorno);
94      //-------------------------------------------------------
95
```

```
96      //tabela_registros_substituidor.cpp ------------------------
97      void substituaIntervalos(const string &atributo, ValorBase *const
            valorNovo, const vector<Intervalo> &intervalos, const bool
            missingsTmb, const bool invalidosTmb);
98      void substituaIntervalosDP(const string &atributo, const Distribuicao &
            distribuicao, const vector<Intervalo> &intervalos, const bool
            missingsTmb, const bool invalidosTmb);
99      void substituaIntervalosDPPorIndice(const string &atributo, const
            vector<size_t> &quaisIndices, const Distribuicao &distribuicao,
            const vector<Intervalo> &intervalos, const bool missingsTmb, const
             bool invalidosTmb);
100     void substituaIntervalosRLS(const string &atributoInd, const string &
            atributoDep, const RegressaoLinearSimples &regressao, const vector
            <Intervalo> &intervalos, const bool missingsTmb, const bool
            invalidosTmb);
101     void substituaIntervalosRLSPorIndice(const string &atributoInd, const
            string &atributoDep, const vector<size_t> &quaisIndices, const
            RegressaoLinearSimples &regressao, const vector<Intervalo> &
            intervalos, const bool missingsTmb, const bool invalidosTmb);
102     void substituaInvalidos(const string &atributo, ValorBase *const
            valorNovo);
103     void substituaInvalidosDP(const string &atributo, const Distribuicao &
            distribuicao);
104     void substituaInvalidosDPPorIndice(const string &atributo, const vector
            <size_t> &quaisIndices, const Distribuicao &distribuicao);
105     void substituaInvalidosRLS(const string &atributoInd, const string &
            atributoDep, const RegressaoLinearSimples &regressao);
106     void substituaInvalidosRLSPorIndice(const string &atributoInd, const
            string &atributoDep, const vector<size_t> &quaisIndices, const
            RegressaoLinearSimples &regressao);
107     void substituaLocalizados(const string &atributo, const vector<
            ValorBase *const> &antigosValores, ValorBase *const novoValor,
            const bool missingsTmb, bool invalidosTmb);
108     void substituaLocalizadosPorIndice(const string &atributo, const vector
            <size_t> &quaisIndices, const vector<ValorBase *const> &
            antigosValores, ValorBase *const novoValor, const bool missingsTmb
            , bool invalidosTmb);
109     void substituaMissings(const string &atributo, ValorBase *const
            valorNovo);
110     void substituaMissingsDP(const string &atributo, const Distribuicao &
            distribuicao);
111     void substituaMissingsDPPorIndice(const string &atributo, const vector<
            size_t> &quaisIndices, const Distribuicao &distribuicao);
112     void substituaMissingsRLS(const string &atributoInd, const string &
            atributoDep, const RegressaoLinearSimples &regressao);
113     void substituaMissingsRLSPorIndice(const string &atributoInd, const
            string &atributoDep, const vector<size_t> &quaisIndices, const
            RegressaoLinearSimples &regressao);
114     void substituaTodos(const string &atributo, ValorBase *const valorNovo)
            ;
115     void substituaTodosDP(const string &atributo, const Distribuicao &
            distribuicao);
116     void substituaTodosDPPorIndice(const string &atributo, const vector<
            size_t> &quaisIndices, const Distribuicao &distribuicao);
117     void substituaTodosPorIndice(const string &atributo, ValorBase *const
            valorNovo, const vector<size_t> &quaisIndices);
118     void substituaTodosRLS(const string &atributoInd, const string &
            atributoDep, const RegressaoLinearSimples &regressao);
119     void substituaTodosRLSPorIndice(const string &atributoInd, const string
             &atributoDep, const vector<size_t> &quaisIndices, const
            RegressaoLinearSimples &regressao);
120     //--------------------------------------------------------------
121
122 private:
123     static TabelaDeRegistros *instancia;
124     MapRegistros _registros;
125     vector<string> _atributos;
126     vector<TipoDado_AT> _tipos;
127     size_t _totalRegistros;
128     size_t _novoRegistroIndice;
129
130     TabelaDeRegistros();
131     TabelaDeRegistros(const TabelaDeRegistros&){};
132     TabelaDeRegistros &operator =(const TabelaDeRegistros&){};
133
134     ~TabelaDeRegistros();
135
136     bool pertenceAosIntervalos(const vector<TabelaDeRegistros::Intervalo> &
            intervalos, const double valorDbl) const;
137 };
138 #endif
```

Listagem B.121: tabela_registros.h

```
1  #include "tabela_registros.h"
2
3  #include "utilidades.h"
4
5  const Registro &TabelaDeRegistros::registro(const size_t indiceRegistro)
       const
6  {
7      return _registros.at(indiceRegistro);
8  }
9
10 size_t TabelaDeRegistros::novoRegistro()
11 {
12     _totalRegistros++;
13
14     _registros[_novoRegistroIndice] = Registro();
15     Registro &r = _registros[_novoRegistroIndice];
16
17     for (vector<string>::const_iterator citAtributo = _atributos.cbegin();
18          citAtributo != _atributos.end();
19          citAtributo++)
20     {
21         const string &atributo = *citAtributo;
22
23         ValorBase *const valor = gerarValor(atributo, "");
24
25         r.insiraValor(valor);
26
27         delete valor;
28     }
29
30     _novoRegistroIndice++;
31
32     return _novoRegistroIndice - 1;
33 }
34
35 void TabelaDeRegistros::atualizeRegistro(
36     const size_t indiceRegistro,
37     const string &atributo,
38     ValorBase *const novoValor)
39 {
40     const size_t indiceAtributo = this->indiceAtributo(atributo);
41
42     MapRegistros::iterator itRegistro = _registros.find(indiceRegistro);
43     if (itRegistro != _registros.cend())
44     {
45         Registro &registro = itRegistro->second;
46         registro.substituaValor(indiceAtributo, novoValor);
47     }
48 }
49
50 void TabelaDeRegistros::indiceRegistrosComplementares(
51     vector<size_t> &indicesRegistros,
52     vector<size_t> &indicesComplementaresRetorno) const
53 {
54     indicesComplementaresRetorno.clear();
55
56     std::sort(indicesRegistros.begin(), indicesRegistros.end()); //
           difference precisa estar ordenado
57
58     vector<size_t> todosIndices = vector<size_t>();
59     todosIndicesRegistros(todosIndices);
60     std::sort(todosIndices.begin(), todosIndices.end()); //difference
           precisa estar ordenado
61
62     set_difference(todosIndices.begin(),
63         todosIndices.end(),
64         indicesRegistros.begin(),
65         indicesRegistros.end(),
66         inserter(indicesComplementaresRetorno, indicesComplementaresRetorno
               .begin()));
67 }
68
69 void TabelaDeRegistros::indiceRegistrosComplementares(
70     vector<vector<size_t>> &indicesRegistrosColunas,
71     vector<vector<size_t>> &indicesComplementaresColunasRetorno) const
72 {
```

```
73      vector<size_t> todosIndices = vector<size_t>();
74      todosIndicesRegistros(todosIndices);
75      std::sort(todosIndices.begin(), todosIndices.end()); //difference
            precisa estar ordenado
76
77      indicesComplementaresColunasRetorno.clear();
78
79      for (vector<vector<size_t>>::iterator itIndicesRegistros =
            indicesRegistrosColunas.begin();
80          itIndicesRegistros != indicesRegistrosColunas.end();
81          itIndicesRegistros++)
82      {
83          vector<size_t> &indicesRegistros = *itIndicesRegistros;
84          std::sort(indicesRegistros.begin(), indicesRegistros.end()); //
                difference precisa estar ordenado
85
86          indicesComplementaresColunasRetorno.push_back(vector<size_t>());
87          vector<size_t> &indicesComplementaresRetorno =
                indicesComplementaresColunasRetorno.back();
88
89          set_difference(todosIndices.begin(),
90              todosIndices.end(),
91              indicesRegistros.begin(),
92              indicesRegistros.end(),
93              inserter(indicesComplementaresRetorno,
                    indicesComplementaresRetorno.begin()));
94      }
95  }
96
97  void TabelaDeRegistros::registrosPorIndice(const vector<size_t> indices,
        vector<const Registro *const> &registrosRetorno) const
98  {
99      registrosRetorno.reserve(indices.size());
100
101     for (vector<size_t>::const_iterator citIndice = indices.cbegin();
102         citIndice != indices.cend();
103         citIndice++)
104     {
105         const size_t indice = *citIndice;
106         const Registro &registro = _registros.at(indice);
107         registrosRetorno.push_back(&registro);
108     }
109 }
110
111 void TabelaDeRegistros::registrosVetorStrPorIndice(const vector<size_t>
        indices, vector<vector<string>> &registrosRetorno, const bool comId)
        const
112 {
113     registrosRetorno.reserve(indices.size());
114
115     if (comId)
116     {
117         for (vector<size_t>::const_iterator citIndice = indices.cbegin();
118             citIndice != indices.cend();
119             citIndice++)
120         {
121             const size_t indice = *citIndice;
122             const Registro &registro = _registros.at(indice);
123             vector<string> campos = registro.valoresString();
124             campos.insert(campos.begin(), Utilidades::dblToStdStr(indice,
                    0));
125             registrosRetorno.push_back(campos);
126         }
127     }
128     else
129     {
130         for (vector<size_t>::const_iterator citIndice = indices.cbegin();
131             citIndice != indices.cend();
132             citIndice++)
133         {
134             const size_t indice = *citIndice;
135             const Registro &registro = _registros.at(indice);
136             registrosRetorno.push_back(registro.valoresString());
137         }
138     }
139 }
140
141 void TabelaDeRegistros::removaRegistro(const size_t indiceRegistro)
142 {
143     MapRegistros::const_iterator citRegistro = _registros.find(
            indiceRegistro);
144     if (citRegistro != _registros.cend())
```

```
145     {
146         _registros.erase(citRegistro);
147         _totalRegistros--;
148     }
149 }
150
151 void TabelaDeRegistros::todosRegistros(vector<const Registro *const> &
        registrosRetornados) const
152 {
153     registrosRetornados.reserve(_totalRegistros);
154
155     for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
156         citRegistro != _registros.cend();
157         citRegistro++)
158     {
159         const Registro &registro = citRegistro->second;
160         registrosRetornados.push_back(&registro);
161     }
162 }
163
164 void TabelaDeRegistros::todosRegistrosVetorStr(vector<vector<string>> &
        registrosRetorno, const bool comId) const
165 {
166     registrosRetorno.reserve(_totalRegistros);
167
168     if (comId)
169     {
170         for (MapRegistros::const_iterator citRegistro = _registros.cbegin()
                ;
171             citRegistro != _registros.cend();
172             citRegistro++)
173         {
174             const Registro &registro = citRegistro->second;
175             const size_t indiceRegistro = citRegistro->first;
176
177             vector<string> campos = registro.valoresString();
178             campos.insert(campos.begin(), Utilidades::dblToStdStr(
                    indiceRegistro, 0));
179             registrosRetorno.push_back(campos);
180         }
181     }
182     else
183     {
184         for (MapRegistros::const_iterator citRegistro = _registros.cbegin()
                ;
185             citRegistro != _registros.cend();
186             citRegistro++)
187         {
188             const Registro &registro = citRegistro->second;
189             registrosRetorno.push_back(registro.valoresString());
190         }
191     }
192 }
```

Listagem B.122: tabela_registros_get_set_registros.cpp

```
1 #include "tabela_registros.h"
2
3 #include <set>
4
5 void TabelaDeRegistros::valoresAtributo(
6     const string &atributo,
7     vector<ValorBase *const> &valoresRetorno) const
8 {
9     valoresRetorno.reserve(_totalRegistros);
10
11     const size_t indiceAtributo = this->indiceAtributo(atributo);
12
13     for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
14         citRegistro != _registros.cend();
15         citRegistro++)
16     {
17         const Registro &registro = citRegistro->second;
18         valoresRetorno.push_back(registro.retorneValor(indiceAtributo));
19     }
20 }
21
22 void TabelaDeRegistros::valoresAtributoQuant(
23     const string &atributo,
```

```
24      const bool todosMissingsInvalidos,
25      vector<ValorQuantitativo *const> &valoresRetorno) const
26  {
27      valoresRetorno.reserve(_totalRegistros);
28
29      const size_t indiceAtributo = this->indiceAtributo(atributo);
30
31      for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
32          citRegistro != _registros.cend();
33          citRegistro++)
34      {
35          const Registro &registro = citRegistro->second;
36          ValorQuantitativo *const valor = registro.retorneValorQuant(
                indiceAtributo);
37
38          if ((!valor->missing() && valor->formatoValido()) ||
                todosMissingsInvalidos)
39          {
40              valoresRetorno.push_back(valor);
41          }
42      }
43  }
44
45  void TabelaDeRegistros::valoresAtributoQuantPorIndice(
46          const string &atributo,
47          const vector<size_t> &quaisIndices,
48          vector<ValorQuantitativo *const> &valoresRetorno) const
49  {
50      const size_t indiceAtributo = this->indiceAtributo(atributo);
51
52      valoresRetorno.reserve(quaisIndices.size());
53
54      for (vector<size_t>::const_iterator indice = quaisIndices.cbegin();
55          indice != quaisIndices.cend();
56          indice++)
57      {
58          if (_registros.count(*indice) != 0)
59          {
60              const Registro &registro = _registros.at(*indice);
61              valoresRetorno.push_back(registro.retorneValorQuant(
                    indiceAtributo));
62          }
63      }
64  }
65
66  void TabelaDeRegistros::valoresAtributoQuantDbl(
67      const string &atributo,
68      const bool todosMissingsInvalidos,
69      vector<double> &valoresRetorno)
70  {
71      valoresRetorno.reserve(_totalRegistros);
72
73      const size_t indiceAtributo = this->indiceAtributo(atributo);
74
75      for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
76          citRegistro != _registros.cend();
77          citRegistro++)
78      {
79          const Registro &registro = citRegistro->second;
80
81          ValorQuantitativo *const valor = registro.retorneValorQuant(
                indiceAtributo);
82          if ((!valor->missing() && valor->formatoValido()) ||
                todosMissingsInvalidos)
83          {
84              valoresRetorno.push_back(valor->valorDbl());
85          }
86      }
87  }
88
89  void TabelaDeRegistros::valoresAtributosQuantDbl(
90      const vector<string> &atributos,
91      const bool todosMissingsInvalidos,
92      vector<vector<double>> &valoresRetorno) const
93  {
94      valoresRetorno.reserve(atributos.size());
95
96      vector<size_t> indicesAtributos = vector<size_t>();
97      indicesAtributos.reserve(atributos.size());
98
99      for (vector<string>::const_iterator citAtributo = atributos.cbegin();
100         citAtributo != atributos.cend();
```

```
101         citAtributo++)
102     {
103         const string &atributo = *citAtributo;
104 
105         indicesAtributos.push_back(this->indiceAtributo(atributo));
106         valoresRetorno.push_back(vector<double>());
107         valoresRetorno.back().reserve(_totalRegistros);
108     }
109 
110     for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
111         citRegistro != _registros.cend();
112         citRegistro++)
113     {
114         const Registro &registro = citRegistro->second;
115 
116         //se alguns dos valores do registro for missing ou invalido e nao
                for pra considerar todos
117         if (!todosMissingsInvalidos && registro.
                algumValorInvalidoOuMissingNosIndices(indicesAtributos))
118         {
119             continue;
120         }
121 
122         size_t i = 0;
123         for (vector<size_t>::const_iterator citIndiceAtributo =
                indicesAtributos.cbegin();
124             citIndiceAtributo != indicesAtributos.cend();
125             citIndiceAtributo++, i++)
126         {
127             const size_t indiceAtributo = *citIndiceAtributo;
128 
129             ValorQuantitativo *const valor = registro.retorneValorQuant(
                    indiceAtributo);
130 
131             if ((!valor->missing() && valor->formatoValido()) ||
                    todosMissingsInvalidos)
132             {
133                 valoresRetorno[i].push_back(valor->valorDbl());
134             }
135         }
136     }
137 }
138 
139 void TabelaDeRegistros::valoresAtributosQuantDblPorIndice(
140     const vector<string> &atributos,
141     const vector<size_t> &quaisIndices,
142     const bool todosMissingsInvalidos,
143     vector<vector<double>> &valoresRetorno) const
144 {
145     valoresRetorno.reserve(atributos.size());
146 
147     vector<size_t> indicesAtributos = vector<size_t>();
148     indicesAtributos.reserve(atributos.size());
149 
150     //melhor tempo de acesso
151     set<size_t> quaisIndicesSet = set<size_t>(quaisIndices.cbegin(),
            quaisIndices.cend());
152 
153     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
154         citAtributo != atributos.cend();
155         citAtributo++)
156     {
157         const string &atributo = *citAtributo;
158 
159         indicesAtributos.push_back(this->indiceAtributo(atributo));
160         valoresRetorno.push_back(vector<double>());
161         valoresRetorno.back().reserve(quaisIndices.size());
162     }
163 
164     for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
165         citRegistro != _registros.cend();
166         citRegistro++)
167     {
168         const size_t indiceRegistro = citRegistro->first;
169         if (quaisIndicesSet.find(indiceRegistro) == quaisIndicesSet.cend())
170         {
171             continue;
172         }
173 
174         const Registro &registro = citRegistro->second;
175 
176         //se alguns dos valores do registro for missing ou invalido e nao
```

```
                for pra considerar todos
177         if (!todosMissingsInvalidos && registro.
                algumValorInvalidoOuMissingNosIndices(indicesAtributos))
178         {
179             continue;
180         }
181
182         size_t i = 0;
183         for (vector<size_t>::const_iterator citIndiceAtributo =
                indicesAtributos.cbegin();
184             citIndiceAtributo != indicesAtributos.cend();
185             citIndiceAtributo++, i++)
186         {
187             const size_t indiceAtributo = *citIndiceAtributo;
188
189             ValorQuantitativo *const valor = registro.retorneValorQuant(
                    indiceAtributo);
190
191             if ((!valor->missing() && valor->formatoValido()) ||
                    todosMissingsInvalidos)
192             {
193                 valoresRetorno[i].push_back(valor->valorDbl());
194             }
195         }
196     }
197 }
198
199 void TabelaDeRegistros::valoresAtributosQuantDblPorIndicePorAtributo(
200     const vector<string> &atributos,
201     const vector<vector<size_t>> &quaisIndicesPorAtributo,
202     vector<vector<double>> &valoresRetorno) const
203 {
204     valoresRetorno.reserve(atributos.size());
205
206     size_t i = 0;
207     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
208         citAtributo != atributos.cend();
209         citAtributo++, i++)
210     {
211         valoresRetorno.push_back(vector<double>());
212         valoresRetorno.back().reserve(quaisIndicesPorAtributo[i].size());
213     }
214
215     i = 0;
216     for (vector<string>::const_iterator citAtributo = atributos.cbegin();
217         citAtributo != atributos.cend();
218         citAtributo++, i++)
219     {
220         const string &atributo = *citAtributo;
221         const size_t indiceAtributo = this->indiceAtributo(atributo);
222         const vector<size_t> &quaisIndices = quaisIndicesPorAtributo[i];
223
224         for (vector<size_t>::const_iterator citIndiceReg = quaisIndices.
                cbegin();
225             citIndiceReg != quaisIndices.cend();
226             citIndiceReg++)
227         {
228             const size_t indiceReg = *citIndiceReg;
229             const Registro &registro = _registros.at(indiceReg);
230
231             ValorQuantitativo *const valor = registro.retorneValorQuant(
                    indiceAtributo);
232
233             if (!valor->missing() && valor->formatoValido())
234             {
235                 valoresRetorno[i].push_back(valor->valorDbl());
236             }
237         }
238     }
239 }
240
241 void TabelaDeRegistros::valoresAtributoStr(const string &atributo, map<
        size_t, string> &valoresRetorno) const
242 {
243     const size_t indiceAtributo = this->indiceAtributo(atributo);
244
245     for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
246         citRegistro != _registros.cend();
247         citRegistro++)
248     {
249         const size_t indiceRegistro = citRegistro->first;
250         const Registro &registro = citRegistro->second;
```

```
251
252             valoresRetorno[indiceRegistro] = registro.retorneValor(
                    indiceAtributo)->valorStr();
253         }
254     }
255
256     void TabelaDeRegistros::valoresAtributoStrPorIndice(
257         const string &atributo,
258         const vector<size_t> &quaisIndices,
259         map<size_t, string> &valoresRetorno) const
260     {
261         const size_t indiceAtributo = this->indiceAtributo(atributo);
262
263         for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
264             citIndice != quaisIndices.cend();
265             citIndice++)
266         {
267             const size_t indice = *citIndice;
268             const Registro &registro = _registros.at(indice);
269
270             valoresRetorno[indice] = registro.retorneValor(indiceAtributo)->
                    valorStr();
271         }
272     }
```

Listagem B.123: tabela_registros_get_valores.cpp

```
1   #include "tabela_registros.h"
2
3   void TabelaDeRegistros::identificarRegistrosMissingsInvalidos(
4       vector<size_t> &registrosMissInvaRetorno,
5       map<string, vector<size_t>> &missInvPorAtribRetorno) const
6   {
7       const vector<size_t> indicesAtributos = this->indicesAtributos(
            _atributos);
8
9       for (vector<string>::const_iterator citAtributo = _atributos.cbegin();
10          citAtributo != _atributos.cend();
11          citAtributo++)
12      {
13          const string &atributo = *citAtributo;
14          missInvPorAtribRetorno[atributo] = vector<size_t>();
15      }
16
17      for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
18          citRegistro != _registros.cend();
19          citRegistro++)
20      {
21          const size_t indiceRegistro = citRegistro->first;
22          const Registro &registro = citRegistro->second;
23
24          bool registroTemMissInv = false;
25
26          for (vector<size_t>::const_iterator citIdxAtributo =
                indicesAtributos.cbegin();
27              citIdxAtributo != indicesAtributos.cend();
28              citIdxAtributo++)
29          {
30              const size_t indiceAtributo = *citIdxAtributo;
31              const string &atributo = _atributos[indiceAtributo];
32
33              ValorBase *const valor = registro.retorneValor(indiceAtributo);
34              ValorQuantitativo *const valorQuant = registro.
                    retorneValorQuant(indiceAtributo);
35              if (valor->missing() || (valorQuant != NULL && !valorQuant->
                    formatoValido()))
36              {
37                  missInvPorAtribRetorno[atributo].push_back(indiceRegistro);
38                  registroTemMissInv = true;
39              }
40          }
41
42          if (registroTemMissInv)
43          {
44              registrosMissInvaRetorno.push_back(indiceRegistro);
45          }
46      }
47  }
48
```

```
49 void TabelaDeRegistros::identificarRegistrosPorValor(
50     const string &atributo,
51     const vector<ValorBase *const> &valores,
52     const bool missingsTmb,
53     bool invalidosTmb,
54     vector<size_t> &indicesIdentificadosRetorno) const
55 {
56     const size_t indiceAtributo = this->indiceAtributo(atributo);
57
58     if (ehQualitativo(atributo))
59     {
60         invalidosTmb = false;
61     }
62
63     for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
64         citRegistro != _registros.cend();
65         citRegistro++)
66     {
67         const Registro &registro = citRegistro->second;
68         const size_t indiceRegistro = citRegistro->first;
69         ValorBase *const valorDados = registro.retorneValor(indiceAtributo)
                ;
70
71         bool encontrou = false;
72
73         for (vector<ValorBase *const>::const_iterator citValorBusca =
                valores.begin();
74             citValorBusca != valores.cend() && !encontrou;
75             citValorBusca++)
76         {
77             ValorBase *const valorBusca = *citValorBusca;
78             if (*valorBusca == *valorDados || missingsTmb && valorDados->
                    missing() || invalidosTmb && !dynamic_cast<
                    ValorQuantitativo *const>(valorDados)->formatoValido())
79             {
80                 encontrou = true;
81             }
82         }
83
84         if (encontrou)
85         {
86             indicesIdentificadosRetorno.push_back(indiceRegistro);
87         }
88     }
89 }
90
91 void TabelaDeRegistros::identificarRegistrosQuantPorIntervalo(
92     const string &atributo,
93     const vector<TabelaDeRegistros::Intervalo> &intervalos,
94     const bool missingsTmb,
95     const bool invalidosTmb,
96     vector<size_t> &indicesIdentificadosRetorno) const
97 {
98     const size_t indiceAtributo = this->indiceAtributo(atributo);
99
100    for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
101        citRegistro != _registros.cend();
102        citRegistro++)
103    {
104        const size_t indiceRegistro = citRegistro->first;
105        const Registro &registro = citRegistro->second;
106
107        ValorQuantitativo *const valor = registro.retorneValorQuant(
                indiceAtributo);
108        if (pertenceAosIntervalos(intervalos, valor->valorDbl()) ||
                missingsTmb && valor->missing() || invalidosTmb && !valor->
                formatoValido())
109        {
110            indicesIdentificadosRetorno.push_back(indiceRegistro);
111        }
112    }
113 }
114
115 void TabelaDeRegistros::identificarRegistrosQuantPorIntervaloPorIndice(
116     const string &atributo,
117     const vector<size_t> &quaisIndices,
118     const vector<TabelaDeRegistros::Intervalo> &intervalos,
119     const bool missingsTmb,
120     const bool invalidosTmb,
121     vector<size_t> &indicesIdentificadosRetorno) const
122 {
123     const size_t indiceAtributo = this->indiceAtributo(atributo);
```

```
124
125     for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
126         citIndice != quaisIndices.cend();
127         citIndice++)
128     {
129         const size_t indice = *citIndice;
130         const Registro &registro = _registros.at(indice);
131
132         ValorQuantitativo *const valor = registro.retorneValorQuant(
                indiceAtributo);
133         if (pertenceAosIntervalos(intervalos, valor->valorDbl()) ||
                missingsTmb && valor->missing() || invalidosTmb && !valor->
                formatoValido())
134         {
135             indicesIdentificadosRetorno.push_back(indice);
136         }
137     }
138 }
139
140 void TabelaDeRegistros::identificarRegistrosMultiAtributos(
141     const vector<string> &atribsQuant,
142     const vector<vector<Intervalo>> &intervalos,
143     const vector<bool> &missingsQuantTmb,
144     const vector<bool> &invalidosQuantTmb,
145     const vector<string> &atribsQual,
146     const vector<vector<ValorBase *const>> &valores,
147     const vector<bool> &missingsQualTmb,
148     const vector<bool> &invalidosQualTmb,
149     vector<size_t> &indicesIdentificadosRetorno) const
150 {
151     bool temQuant = !atribsQuant.empty();
152     bool temQual = !atribsQual.empty();
153
154     indicesIdentificadosRetorno.clear();
155
156     vector<size_t> identificadosQuant = vector<size_t>();
157
158     if (temQuant)
159     {
160         identificarRegistrosQuantPorIntervalo(atribsQuant.front(),
                intervalos.front(), missingsQuantTmb.front(),
                invalidosQuantTmb.front(), identificadosQuant);
161         std::sort(identificadosQuant.begin(), identificadosQuant.end()); //
                intersection precisa estar ordenado
162
163         if (identificadosQuant.empty()) {return;} //se um é vazio a
                intersecção é vazia
164
165         for (size_t i = 1; i < atribsQuant.size(); i++) //+1 pois o
                primeiro ja foi acima
166         {
167             vector<size_t> identificadosAtual = vector<size_t>();
168             identificarRegistrosQuantPorIntervalo(atribsQuant[i],
                    intervalos[i], missingsQuantTmb[i], invalidosQuantTmb[i],
                    identificadosAtual);
169             std::sort(identificadosAtual.begin(), identificadosAtual.end())
                    ; //intersection precisa estar ordenado
170
171             if (identificadosAtual.empty()) {return;} //se um é vazio a
                    intersecção é vazia
172
173             //temporario vai guardar o resultado da interseccao
174             vector<size_t> identificadosQuantTemp = vector<size_t>();
175
176             set_intersection(identificadosQuant.begin(),
177                 identificadosQuant.end(),
178                 identificadosAtual.begin(),
179                 identificadosAtual.end(),
180                 inserter(identificadosQuantTemp, identificadosQuantTemp.
                        begin()));
181
182             identificadosQuant = identificadosQuantTemp;
183             std::sort(identificadosQuant.begin(), identificadosQuant.end())
                    ; //intersection precisa estar ordenado
184         }
185     }
186
187     vector<size_t> identificadosQual = vector<size_t>();
188
189     if (temQual)
190     {
191         identificarRegistrosPorValor(atribsQual.front(), valores.front(),
```

```
                missingsQualTmb.front(), invalidosQualTmb.front(),
                identificadosQual);
192         std::sort(identificadosQual.begin(), identificadosQual.end()); //
                intersection precisa estar ordenado
193
194         if (identificadosQual.empty()) {return;} //se um é vazio a
                intersecção é vazia
195
196         for (size_t i = 1; i < atribsQual.size(); i++) //+1 pois o primeiro
                ja foi acima
197         {
198             vector<size_t> identificadosAtual = vector<size_t>();
199             identificarRegistrosPorValor(atribsQual[i], valores[i],
                    missingsQualTmb[i], invalidosQualTmb[i],
                    identificadosAtual);
200             std::sort(identificadosAtual.begin(), identificadosAtual.end())
                    ; //intersection precisa estar ordenado
201
202             if (identificadosAtual.empty()) {return;} //se um é vazio a
                    intersecção é vazia
203
204             //temporario vai guardar o resultado da interseccao
205             vector<size_t> identificadosQualTemp = vector<size_t>();
206
207             set_intersection(identificadosQual.begin(),
208                 identificadosQual.end(),
209                 identificadosAtual.begin(),
210                 identificadosAtual.end(),
211                 inserter(identificadosQualTemp, identificadosQualTemp.begin
                        ()));
212
213             identificadosQual = identificadosQualTemp;
214             std::sort(identificadosQual.begin(), identificadosQual.end());
                    //intersection precisa estar ordenado
215         }
216     }
217
218     if (temQuant && temQual)
219     {
220         set_intersection(identificadosQual.begin(),
221             identificadosQual.end(),
222             identificadosQuant.begin(),
223             identificadosQuant.end(),
224             inserter(indicesIdentificadosRetorno,
                    indicesIdentificadosRetorno.begin()));
225     }
226     else if (temQuant)
227     {
228         indicesIdentificadosRetorno = identificadosQuant;
229     }
230     else if (temQual)
231     {
232         indicesIdentificadosRetorno = identificadosQual;
233     }
234 }
```

Listagem B.124: tabela_registros_identificador.cpp

```
1  #include "tabela_registros.h"
2
3  void TabelaDeRegistros::popularColunasDP(
4      const vector<string> &atributos,
5      const vector<const Distribuicao> &distribuicoes)
6  {
7      for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
8          citRegistro != _registros.cend();
9          citRegistro++)
10     {
11         const size_t indiceRegistro = citRegistro->first;
12         popularRegistroDP(indiceRegistro, atributos, distribuicoes);
13     }
14 }
15
16 void TabelaDeRegistros::popularColunasRLS(
17     const vector<string> &atributosIndep,
18     const vector<string> &atributosDep,
19     const vector<const RegressaoLinearSimples> &regressoes)
20 {
21     for (MapRegistros::const_iterator citRegistro = _registros.cbegin();
```

```
22          citRegistro != _registros.cend();
23          citRegistro++)
24      {
25          const size_t indiceRegistro = citRegistro->first;
26          popularRegistroRLS(indiceRegistro, atributosIndep, atributosDep,
                regressoes);
27      }
28  }
29
30  void TabelaDeRegistros::popularRegistroDP(
31      const size_t indiceRegistro,
32      const vector<string> &atributos,
33      const vector<const Distribuicao> &distribuicoes)
34  {
35      Registro &registro = _registros[indiceRegistro];
36
37      size_t i = 0;
38      for (vector<string>::const_iterator citAtributo = atributos.cbegin();
39          citAtributo != atributos.cend();
40          citAtributo++, i++)
41      {
42          const string &atributo = *citAtributo;
43          const size_t indiceAtributo = this->indiceAtributo(atributo);
44
45          ValorBase *novoValor = gerarValorQuant(distribuicoes[i].
                gerarValorAleatorio());
46          registro.substituaValor(indiceAtributo, novoValor);
47          delete novoValor;
48      }
49
50  }
51
52  void TabelaDeRegistros::popularRegistroRLS(
53      const size_t indiceRegistro,
54      const vector<string> &atributosIndep,
55      const vector<string> &atributosDep,
56      const vector<const RegressaoLinearSimples> &regressoes)
57  {
58      //copia, para os valores independentes
59      Registro registroCopia = _registros[indiceRegistro];
60
61      //original, para se alterar
62      Registro &registroOriginal = _registros[indiceRegistro];
63
64      const size_t total = atributosDep.size();
65
66      for (size_t i = 0; i < total; i++)
67      {
68          const string &atributoInde = atributosIndep[i];
69          const string &atribitoDep = atributosDep[i];
70          const RegressaoLinearSimples &regressao = regressoes[i];
71
72          const size_t indiceAtributoInd = this->indiceAtributo(atributoInde)
                ;
73          const size_t indiceAtributoDep = this->indiceAtributo(atribitoDep);
74
75          if (registroCopia.retorneValorQuant(indiceAtributoInd)->missing()
                || !registroCopia.retorneValorQuant(indiceAtributoInd)->
                formatoValido())
76          {
77              continue;
78          }
79
80          const double valorIndependente = registroCopia.retorneValorQuant(
                indiceAtributoInd)->valorDbl();
81          const double valorDependente = regressao.gerarValorDependente(
                valorIndependente);
82
83          ValorBase *novoValor = gerarValorQuant(valorDependente);
84          registroOriginal.substituaValor(indiceAtributoDep, novoValor);
85          delete novoValor;
86      }
87  }
```

Listagem B.125: tabela_registros_populador.cpp

```
1  #include "tabela_registros.h"
2
3  void TabelaDeRegistros::removaRegistrosPorValores(
```

```
4       const string &atributo,
5       const vector<ValorBase *const> &valores,
6       const bool missingsTmb,
7       const bool invalidosTmb,
8       vector<size_t> &indicesRemovidosRetorno)
9   {
10      identificarRegistrosPorValor(atributo, valores, missingsTmb,
            invalidosTmb, indicesRemovidosRetorno);
11
12      for (vector<size_t>::const_iterator citIndice = indicesRemovidosRetorno
            .cbegin();
13          citIndice != indicesRemovidosRetorno.cend();
14          citIndice++)
15      {
16          const size_t indice = *citIndice;
17          removaRegistro(indice);
18      }
19  }
20
21  void TabelaDeRegistros::removerRegistrosQuantPorIntervalo(
22      const vector<TabelaDeRegistros::Intervalo> &intervalos,
23      const string &atributo,
24      const bool missingsTmb,
25      const bool invalidosTmb,
26      vector<size_t> &indicesRemovidosRetorno)
27  {
28      identificarRegistrosQuantPorIntervalo(atributo, intervalos, missingsTmb
            , invalidosTmb, indicesRemovidosRetorno);
29
30      for (vector<size_t>::const_iterator citIndice = indicesRemovidosRetorno
            .cbegin();
31          citIndice != indicesRemovidosRetorno.cend();
32          citIndice++)
33      {
34          const size_t indice = *citIndice;
35          removaRegistro(indice);
36      }
37  }
```

Listagem B.126: tabela_registros_removedor.cpp

```
1   #include "tabela_registros.h"
2
3   void TabelaDeRegistros::substituaLocalizados(
4       const string &atributo,
5       const vector<ValorBase *const> &antigosValores,
6       ValorBase *const novoValor,
7       const bool missingsTmb,
8       bool invalidosTmb)
9   {
10      const size_t indiceAtributo = this->indiceAtributo(atributo);
11
12      if (ehQualitativo(atributo))
13      {
14          invalidosTmb = false;
15      }
16
17      for (MapRegistros::iterator itRegistro = _registros.begin();
18          itRegistro != _registros.end();
19          itRegistro++)
20      {
21          Registro &registro = itRegistro->second;
22
23          ValorBase *const valorAtual = registro.retorneValor(indiceAtributo)
                ;
24
25          bool encontrou = false;
26          for (vector<ValorBase *const>::const_iterator citValorAntigo =
                antigosValores.cbegin();
27              citValorAntigo != antigosValores.cend() && !encontrou;
28              citValorAntigo++)
29          {
30              ValorBase *const valorAntigo = *citValorAntigo;
31
32              if (*valorAtual == *valorAntigo || missingsTmb && valorAntigo->
                    missing() || invalidosTmb && !dynamic_cast<
                    ValorQuantitativo *const>(valorAntigo)->formatoValido())
33              {
34                  encontrou = true;
```

```
35              }
36          }
37
38          if (encontrou)
39          {
40              registro.substituaValor(indiceAtributo, novoValor);
41          }
42      }
43  }
44
45  void TabelaDeRegistros::substituaLocalizadosPorIndice(
46      const string &atributo,
47      const vector<size_t> &quaisIndices,
48      const vector<ValorBase *const> &antigosValores,
49      ValorBase *const novoValor,
50      const bool missingsTmb,
51      bool invalidosTmb)
52  {
53      const size_t indiceAtributo = this->indiceAtributo(atributo);
54
55      if (ehQualitativo(atributo))
56      {
57          invalidosTmb = false;
58      }
59
60      for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
61          citIndice != quaisIndices.cend();
62          citIndice++)
63      {
64          const size_t indice = *citIndice;
65          Registro &registro = _registros.at(indice);
66
67          ValorBase *const valorAtual = registro.retorneValor(indiceAtributo)
                ;
68
69          bool encontrou = false;
70          for (vector<ValorBase *const>::const_iterator citValorAntigo =
                antigosValores.cbegin();
71              citValorAntigo != antigosValores.cend() && !encontrou;
72              citValorAntigo++)
73          {
74              ValorBase *const valorAntigo = *citValorAntigo;
75
76              if (*valorAtual == *valorAntigo || missingsTmb && valorAntigo->
                    missing() || invalidosTmb && !dynamic_cast<
                    ValorQuantitativo *const>(valorAntigo)->formatoValido())
77              {
78                  encontrou = true;
79              }
80          }
81
82          if (encontrou)
83          {
84              registro.substituaValor(indiceAtributo, novoValor);
85          }
86      }
87  }
88
89  void TabelaDeRegistros::substituaInvalidos(
90      const string &atributo,
91      ValorBase *const valorNovo)
92  {
93      const size_t indiceAtributo = this->indiceAtributo(atributo);
94
95      for (MapRegistros::iterator itRegistro = _registros.begin();
96          itRegistro != _registros.end();
97          itRegistro++)
98      {
99          Registro &registro = itRegistro->second;
100         ValorQuantitativo *const valorAtual = registro.retorneValorQuant(
                indiceAtributo);
101
102         //nao considera os missings, apenas os nao validos
103         if (!valorAtual->missing() && !valorAtual->formatoValido())
104         {
105             registro.substituaValor(indiceAtributo, valorNovo);
106         }
107     }
108 }
109
110 void TabelaDeRegistros::substituaMissings(
111     const string &atributo,
```

```
112     ValorBase *const valorNovo)
113 {
114     const size_t indiceAtributo = this->indiceAtributo(atributo);
115
116     for (MapRegistros::iterator itRegistro = _registros.begin();
117         itRegistro != _registros.end();
118         itRegistro++)
119     {
120         Registro &registro = itRegistro->second;
121         ValorBase *const valorAtual = registro.retorneValor(indiceAtributo)
                ;
122
123         if (valorAtual->missing())
124         {
125             registro.substituaValor(indiceAtributo, valorNovo);
126         }
127     }
128 }
129
130 void TabelaDeRegistros::substituaTodos(
131     const string &atributo,
132     ValorBase *const valorNovo)
133 {
134     const size_t indiceAtributo = this->indiceAtributo(atributo);
135
136     for (MapRegistros::iterator itRegistro = _registros.begin();
137         itRegistro != _registros.end();
138         itRegistro++)
139     {
140         Registro &registro = itRegistro->second;
141         registro.substituaValor(indiceAtributo, valorNovo);
142     }
143 }
144
145 void TabelaDeRegistros::substituaIntervalos(
146     const string &atributo,
147     ValorBase *const valorNovo,
148     const vector<Intervalo> &intervalos,
149     const bool missingsTmb,
150     const bool invalidosTmb)
151 {
152     const size_t indiceAtributo = this->indiceAtributo(atributo);
153
154     vector<size_t> indicesIdentificados = vector<size_t>();
155     identificarRegistrosQuantPorIntervalo(atributo, intervalos, missingsTmb
          , invalidosTmb, indicesIdentificados);
156
157     for (vector<size_t>::const_iterator citIndiceIdentificado =
          indicesIdentificados.cbegin();
158         citIndiceIdentificado != indicesIdentificados.cend();
159         citIndiceIdentificado++)
160     {
161         const size_t indiceIdentificado = *citIndiceIdentificado;
162
163         Registro &registro = _registros[indiceIdentificado];
164         registro.substituaValor(indiceAtributo, valorNovo);
165     }
166 }
167
168 void TabelaDeRegistros::substituaInvalidosDP(const string &atributo, const
       Distribuicao &distribuicao)
169 {
170     const size_t indiceAtributo = this->indiceAtributo(atributo);
171
172     for (MapRegistros::iterator itRegistro = _registros.begin();
173         itRegistro != _registros.end();
174         itRegistro++)
175     {
176         Registro &registro = itRegistro->second;
177         ValorQuantitativo *const valorAtual = registro.retorneValorQuant(
              indiceAtributo);
178
179         //nao remove os missings, apenas os nao validos
180         if (!valorAtual->missing() && !valorAtual->formatoValido())
181         {
182             ValorBase *valorGerado = gerarValorQuant(distribuicao.
                  gerarValorAleatorio());
183             registro.substituaValor(indiceAtributo, valorGerado);
184             delete valorGerado;
185         }
186     }
187 }
```

```
188
189 void TabelaDeRegistros::substituaInvalidosDPPorIndice(const string &
        atributo, const vector<size_t> &quaisIndices, const Distribuicao &
        distribuicao)
190 {
191     const size_t indiceAtributo = this->indiceAtributo(atributo);
192
193     for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
194         citIndice != quaisIndices.cend();
195         citIndice++)
196     {
197         const size_t indice = *citIndice;
198         Registro &registro = _registros.at(indice);
199
200         ValorQuantitativo *const valorAtual = registro.retorneValorQuant(
                indiceAtributo);
201
202         //nao remove os missings, apenas os nao validos
203         if (!valorAtual->missing() && !valorAtual->formatoValido())
204         {
205             ValorBase *valorGerado = gerarValorQuant(distribuicao.
                    gerarValorAleatorio());
206             registro.substituaValor(indiceAtributo, valorGerado);
207             delete valorGerado;
208         }
209     }
210 }
211
212 void TabelaDeRegistros::substituaMissingsDP(const string &atributo, const
        Distribuicao &distribuicao)
213 {
214     const size_t indiceAtributo = this->indiceAtributo(atributo);
215
216     for (MapRegistros::iterator itRegistro = _registros.begin();
217         itRegistro != _registros.end();
218         itRegistro++)
219     {
220         Registro &registro = itRegistro->second;
221         ValorBase *const valorAtual = registro.retorneValor(indiceAtributo)
                ;
222
223         if (registro.retorneValor(indiceAtributo)->missing())
224         {
225             ValorBase *valorGerado = gerarValorQuant(distribuicao.
                    gerarValorAleatorio());
226             registro.substituaValor(indiceAtributo, valorGerado);
227             delete valorGerado;
228         }
229     }
230 }
231
232 void TabelaDeRegistros::substituaMissingsDPPorIndice(const string &atributo
        , const vector<size_t> &quaisIndices, const Distribuicao &distribuicao
        )
233 {
234     const size_t indiceAtributo = this->indiceAtributo(atributo);
235
236     for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
237         citIndice != quaisIndices.cend();
238         citIndice++)
239     {
240         const size_t indice = *citIndice;
241         Registro &registro = _registros.at(indice);
242
243         ValorBase *const valorAtual = registro.retorneValor(indiceAtributo)
                ;
244
245         if (registro.retorneValor(indiceAtributo)->missing())
246         {
247             ValorBase *valorGerado = gerarValorQuant(distribuicao.
                    gerarValorAleatorio());
248             registro.substituaValor(indiceAtributo, valorGerado);
249             delete valorGerado;
250         }
251     }
252 }
253
254 void TabelaDeRegistros::substituaTodosDP(const string &atributo, const
        Distribuicao &distribuicao)
255 {
256     const size_t indiceAtributo = this->indiceAtributo(atributo);
257
```

```
258     for (MapRegistros::iterator itRegistro = _registros.begin();
259         itRegistro != _registros.end();
260         itRegistro++)
261     {
262         Registro &registro = itRegistro->second;
263 
264         ValorBase *valorGerado = gerarValorQuant(distribuicao.
                gerarValorAleatorio());
265         registro.substituaValor(indiceAtributo, valorGerado);
266         delete valorGerado;
267     }
268 }
269 
270 void TabelaDeRegistros::substituaTodosDPPorIndice(const string &atributo,
        const vector<size_t> &quaisIndices, const Distribuicao &distribuicao)
271 {
272     const size_t indiceAtributo = this->indiceAtributo(atributo);
273 
274     for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
275         citIndice != quaisIndices.cend();
276         citIndice++)
277     {
278         const size_t indice = *citIndice;
279         Registro &registro = _registros.at(indice);
280 
281         ValorBase *valorGerado = gerarValorQuant(distribuicao.
                gerarValorAleatorio());
282         registro.substituaValor(indiceAtributo, valorGerado);
283         delete valorGerado;
284     }
285 }
286 
287 void TabelaDeRegistros::substituaIntervalosDP(
288     const string &atributo,
289     const Distribuicao &distribuicao,
290     const vector<Intervalo> &intervalos,
291     const bool missingsTmb,
292     const bool invalidosTmb)
293 {
294     const size_t indiceAtributo = this->indiceAtributo(atributo);
295 
296     vector<size_t> indicesIdentificados = vector<size_t>();
297     identificarRegistrosQuantPorIntervalo(atributo, intervalos, missingsTmb
            , invalidosTmb, indicesIdentificados);
298 
299     for (vector<size_t>::const_iterator citIndiceIdentificado =
            indicesIdentificados.cbegin();
300         citIndiceIdentificado != indicesIdentificados.cend();
301         citIndiceIdentificado++)
302     {
303         const size_t indiceIdentificado = *citIndiceIdentificado;
304         Registro &registro = _registros[indiceIdentificado];
305 
306         ValorBase *valorGerado = gerarValorQuant(distribuicao.
                gerarValorAleatorio());
307         registro.substituaValor(indiceAtributo, valorGerado);
308         delete valorGerado;
309     }
310 }
311 
312 void TabelaDeRegistros::substituaIntervalosDPPorIndice(
313     const string &atributo,
314     const vector<size_t> &quaisIndices,
315     const Distribuicao &distribuicao,
316     const vector<Intervalo> &intervalos,
317     const bool missingsTmb,
318     const bool invalidosTmb)
319 {
320     const size_t indiceAtributo = this->indiceAtributo(atributo);
321 
322     for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
323         citIndice != quaisIndices.cend();
324         citIndice++)
325     {
326         const size_t indice = *citIndice;
327         const Registro &registro = _registros.at(indice);
328 
329     }
330 
331     vector<size_t> indicesIdentificados = vector<size_t>();
332     identificarRegistrosQuantPorIntervaloPorIndice(atributo, quaisIndices,
            intervalos, missingsTmb, invalidosTmb, indicesIdentificados);
```

```
333
334     for (vector<size_t>::const_iterator citIndiceIdentificado =
             indicesIdentificados.cbegin();
335          citIndiceIdentificado != indicesIdentificados.cend();
336          citIndiceIdentificado++)
337     {
338         const size_t indiceIdentificado = *citIndiceIdentificado;
339
340         Registro &registro = _registros[indiceIdentificado];
341
342         ValorBase *valorGerado = gerarValorQuant(distribuicao.
                gerarValorAleatorio());
343         registro.substituaValor(indiceAtributo, valorGerado);
344         delete valorGerado;
345     }
346 }
347
348 void TabelaDeRegistros::substituaInvalidosRLS(const string &atributoInd,
         const string &atributoDep, const RegressaoLinearSimples &regressao)
349 {
350     const size_t indiceAtribDep = this->indiceAtributo(atributoDep);
351     const size_t indiceAtribInd = this->indiceAtributo(atributoInd);
352
353     for (MapRegistros::iterator itRegistro = _registros.begin();
354          itRegistro != _registros.end();
355          itRegistro++)
356     {
357         Registro &registro = itRegistro->second;
358         const size_t indiceRegistro = itRegistro->first;
359
360         ValorQuantitativo *const valorDep = registro.retorneValorQuant(
                indiceAtribDep);
361         ValorQuantitativo *const valorInd = registro.retorneValorQuant(
                indiceAtribInd);
362
363         //nao considera os missings, apenas os nao validos
364         if (!valorDep->missing() && !valorDep->formatoValido()
365             && !valorInd->missing() && valorInd->formatoValido())
366         {
367             const double valorDblInd = valorInd->valorDbl();
368             const double novoValorDblDep = regressao.gerarValorDependente(
                    valorDblInd);
369
370             ValorBase *novoValorDep = gerarValorQuant(novoValorDblDep);
371             registro.substituaValor(indiceAtribDep, novoValorDep);
372             delete novoValorDep;
373         }
374     }
375 }
376
377 void TabelaDeRegistros::substituaInvalidosRLSPorIndice(const string &
         atributoInd, const string &atributoDep, const vector<size_t> &
         quaisIndices, const RegressaoLinearSimples &regressao)
378 {
379     const size_t indiceAtribDep = this->indiceAtributo(atributoDep);
380     const size_t indiceAtribInd = this->indiceAtributo(atributoInd);
381
382     for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
383          citIndice != quaisIndices.cend();
384          citIndice++)
385     {
386         const size_t indice = *citIndice;
387         Registro &registro = _registros.at(indice);
388
389         ValorQuantitativo *const valorDep = registro.retorneValorQuant(
                indiceAtribDep);
390         ValorQuantitativo *const valorInd = registro.retorneValorQuant(
                indiceAtribInd);
391
392         //nao considera os missings, apenas os nao validos
393         if (!valorDep->missing() && !valorDep->formatoValido()
394             && !valorInd->missing() && valorInd->formatoValido())
395         {
396             const double valorDblInd = valorInd->valorDbl();
397             const double novoValorDblDep = regressao.gerarValorDependente(
                    valorDblInd);
398
399             ValorBase *novoValorDep = gerarValorQuant(novoValorDblDep);
400             registro.substituaValor(indiceAtribDep, novoValorDep);
401             delete novoValorDep;
402         }
403     }
```

```
404 }
405
406 void TabelaDeRegistros::substituaMissingsRLS(const string &atributoInd,
        const string &atributoDep, const RegressaoLinearSimples &regressao)
407 {
408     const size_t indiceAtribDep = this->indiceAtributo(atributoDep);
409     const size_t indiceAtribInd = this->indiceAtributo(atributoInd);
410
411     for (MapRegistros::iterator itRegistro = _registros.begin();
412         itRegistro != _registros.end();
413         itRegistro++)
414     {
415         Registro &registro = itRegistro->second;
416         const size_t indiceRegistro = itRegistro->first;
417
418         ValorQuantitativo *const valorDep = registro.retorneValorQuant(
                indiceAtribDep);
419         ValorQuantitativo *const valorInd = registro.retorneValorQuant(
                indiceAtribInd);
420
421         if (valorDep->missing() && !valorInd->missing() && valorInd->
                formatoValido())
422         {
423             const double valorDblInd = valorInd->valorDbl();
424             const double novoValorDblDep = regressao.gerarValorDependente(
                    valorDblInd);
425
426             ValorBase *novoValorDep = gerarValorQuant(novoValorDblDep);
427             registro.substituaValor(indiceAtribDep, novoValorDep);
428             delete novoValorDep;
429         }
430     }
431 }
432
433 void TabelaDeRegistros::substituaMissingsRLSPorIndice(const string &
        atributoInd, const string &atributoDep, const vector<size_t> &
        quaisIndices, const RegressaoLinearSimples &regressao)
434 {
435     const size_t indiceAtribDep = this->indiceAtributo(atributoDep);
436     const size_t indiceAtribInd = this->indiceAtributo(atributoInd);
437
438     for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
439         citIndice != quaisIndices.cend();
440         citIndice++)
441     {
442         const size_t indice = *citIndice;
443         Registro &registro = _registros.at(indice);
444
445         ValorQuantitativo *const valorDep = registro.retorneValorQuant(
                indiceAtribDep);
446         ValorQuantitativo *const valorInd = registro.retorneValorQuant(
                indiceAtribInd);
447
448         if (valorDep->missing() && !valorInd->missing() && valorInd->
                formatoValido())
449         {
450             const double valorDblInd = valorInd->valorDbl();
451             const double novoValorDblDep = regressao.gerarValorDependente(
                    valorDblInd);
452
453             ValorBase *novoValorDep = gerarValorQuant(novoValorDblDep);
454             registro.substituaValor(indiceAtribDep, novoValorDep);
455             delete novoValorDep;
456         }
457     }
458 }
459
460 void TabelaDeRegistros::substituaTodosRLS(const string &atributoInd, const
        string &atributoDep, const RegressaoLinearSimples &regressao)
461 {
462     const size_t indiceAtribDep = this->indiceAtributo(atributoDep);
463     const size_t indiceAtribInd = this->indiceAtributo(atributoInd);
464
465     for (MapRegistros::iterator itRegistro = _registros.begin();
466         itRegistro != _registros.end();
467         itRegistro++)
468     {
469         Registro &registro = itRegistro->second;
470         const size_t indiceRegistro = itRegistro->first;
471
472         ValorQuantitativo *const valorDep = registro.retorneValorQuant(
                indiceAtribDep);
```

```
473             ValorQuantitativo *const valorInd = registro.retorneValorQuant(
                    indiceAtribInd);
474
475             if (!valorInd->missing() && valorInd->formatoValido())
476             {
477                 const double valorDblInd = valorInd->valorDbl();
478                 const double novoValorDblDep = regressao.gerarValorDependente(
                        valorDblInd);
479
480                 ValorBase *novoValorDep = gerarValorQuant(novoValorDblDep);
481                 registro.substituaValor(indiceAtribDep, novoValorDep);
482                 delete novoValorDep;
483             }
484         }
485 }
486
487 void TabelaDeRegistros::substituaTodosRLSPorIndice(const string &
        atributoInd, const string &atributoDep, const vector<size_t> &
        quaisIndices, const RegressaoLinearSimples &regressao)
488 {
489     const size_t indiceAtribDep = this->indiceAtributo(atributoDep);
490     const size_t indiceAtribInd = this->indiceAtributo(atributoInd);
491
492     for (vector<size_t>::const_iterator citIndice = quaisIndices.cbegin();
493         citIndice != quaisIndices.cend();
494         citIndice++)
495     {
496         const size_t indice = *citIndice;
497         Registro &registro = _registros.at(indice);
498
499         ValorQuantitativo *const valorDep = registro.retorneValorQuant(
                indiceAtribDep);
500         ValorQuantitativo *const valorInd = registro.retorneValorQuant(
                indiceAtribInd);
501
502         if (!valorInd->missing() && valorInd->formatoValido())
503         {
504             const double valorDblInd = valorInd->valorDbl();
505             const double novoValorDblDep = regressao.gerarValorDependente(
                    valorDblInd);
506
507             ValorBase *novoValorDep = gerarValorQuant(novoValorDblDep);
508             registro.substituaValor(indiceAtribDep, novoValorDep);
509             delete novoValorDep;
510         }
511     }
512 }
513
514 void TabelaDeRegistros::substituaIntervalosRLS(
515     const string &atributoInd,
516     const string &atributoDep,
517     const RegressaoLinearSimples &regressao,
518     const vector<Intervalo> &intervalos,
519     const bool missingsTmb,
520     const bool invalidosTmb)
521 {
522     vector<size_t> indicesIdentificados = vector<size_t>();
523     identificarRegistrosQuantPorIntervalo(atributoDep, intervalos,
            missingsTmb, invalidosTmb, indicesIdentificados);
524
525     const size_t indiceAtribDep = this->indiceAtributo(atributoDep);
526     const size_t indiceAtribInd = this->indiceAtributo(atributoInd);
527
528     for (vector<size_t>::const_iterator citIndice = indicesIdentificados.
            cbegin();
529         citIndice != indicesIdentificados.cend();
530         citIndice++)
531     {
532         const size_t indice = *citIndice;
533         Registro &registro = _registros.at(indice);
534
535         ValorQuantitativo *const valorDep = registro.retorneValorQuant(
                indiceAtribDep);
536         ValorQuantitativo *const valorInd = registro.retorneValorQuant(
                indiceAtribInd);
537
538         if (!valorInd->missing() && valorInd->formatoValido())
539         {
540             const double valorDblInd = valorInd->valorDbl();
541             const double novoValorDblDep = regressao.gerarValorDependente(
                    valorDblInd);
542
```

```
543             ValorBase *novoValorDep = gerarValorQuant(novoValorDblDep);
544             registro.substituaValor(indiceAtribDep, novoValorDep);
545             delete novoValorDep;
546         }
547     }
548 }
549
550 void TabelaDeRegistros::substituaIntervalosRLSPorIndice(
551     const string &atributoInd,
552     const string &atributoDep,
553     const vector<size_t> &quaisIndices,
554     const RegressaoLinearSimples &regressao,
555     const vector<Intervalo> &intervalos,
556     const bool missingsTmb,
557     const bool invalidosTmb)
558 {
559     vector<size_t> indicesIdentificados = vector<size_t>();
560     identificarRegistrosQuantPorIntervaloPorIndice(atributoDep,
          quaisIndices, intervalos, missingsTmb, invalidosTmb,
          indicesIdentificados);
561
562     const size_t indiceAtribDep = this->indiceAtributo(atributoDep);
563     const size_t indiceAtribInd = this->indiceAtributo(atributoInd);
564
565     for (vector<size_t>::const_iterator citIndice = indicesIdentificados.
          cbegin();
566          citIndice != indicesIdentificados.cend();
567          citIndice++)
568     {
569         const size_t indice = *citIndice;
570         Registro &registro = _registros.at(indice);
571
572         ValorQuantitativo *const valorDep = registro.retorneValorQuant(
                indiceAtribDep);
573         ValorQuantitativo *const valorInd = registro.retorneValorQuant(
                indiceAtribInd);
574
575         if (!valorInd->missing() && valorInd->formatoValido())
576         {
577             const double valorDblInd = valorInd->valorDbl();
578             const double novoValorDblDep = regressao.gerarValorDependente(
                    valorDblInd);
579
580             ValorBase *novoValorDep = gerarValorQuant(novoValorDblDep);
581             registro.substituaValor(indiceAtribDep, novoValorDep);
582             delete novoValorDep;
583         }
584     }
585 }
```

Listagem B.127: tabela_registros_substituidor.cpp

```
1  #ifndef VALORBASE_H
2  #define VALORBASE_H
3
4  #include <string>
5
6  using namespace std;
7
8  class ValorBase
9  {
10 public:
11     bool missing() const {return _missing;}
12     const string &valorStr() const {return _valorStr;}
13     virtual bool operator <(const ValorBase &outro) const = 0;
14     virtual bool operator ==(const ValorBase &outro) const = 0;
15     virtual ValorBase *clone() const = 0;
16 protected:
17     bool _missing;
18     string _valorStr;
19 };
20 #endif
```

Listagem B.128: valor_base.h

```
1 #include "valor_qualitativo.h"
```

```
2
3
4  ValorQualitativo::ValorQualitativo(const string &valor)
5  {
6      _valorStr = valor;
7      _missing = (valor == "") ? true : false;
8  }
9
10
11 ValorQualitativo::~ValorQualitativo()
12 {
13 }
14
15 bool ValorQualitativo::operator ==(const ValorBase &outro) const
16 {
17     const ValorQualitativo *const outraPtr = dynamic_cast<const
           ValorQualitativo *const>(&outro);
18     return (outraPtr != NULL) && (_valorStr == outraPtr->_valorStr);
19 }
20
21 bool ValorQualitativo::operator <(const ValorBase &outro) const
22 {
23     const ValorQualitativo *const outraPtr = dynamic_cast<const
           ValorQualitativo *const>(&outro);
24
25     if (outraPtr == NULL) //comparação entre valores de tipos diferentes,
           qualitativa precede quantitativa
26     {
27         return true;
28     }
29
30     return this->_valorStr < outraPtr->_valorStr;
31 }
32
33 ValorQualitativo* ValorQualitativo::clone() const
34 {
35     return new ValorQualitativo(*this);
36 }
```

Listagem B.129: valor_qualitativo.cpp

```
1  #ifndef VALORQUALITATIVO_H
2  #define VALORQUALITATIVO_H
3
4  #include "valor_base.h"
5
6  using namespace std;
7
8  class ValorQualitativo :
9      public ValorBase
10 {
11 public:
12     ValorQualitativo(const string &valor);
13     ~ValorQualitativo();
14     bool operator <(const ValorBase &outro) const;
15     bool operator ==(const ValorBase &outro) const;
16     ValorQualitativo *clone() const;
17 };
18 #endif
```

Listagem B.130: valor_qualitativo.h

```
1  #include "valor_quantitativo.h"
2  #include "utilidades.h"
3
4  ValorQuantitativo::ValorQuantitativo(const string &valorStr)
5  {
6      _valorDbl = Utilidades::stdStrToDbl(valorStr, true, true, &
           _formatoValido);
7      _missing = (valorStr == "") ? true : false;
8
9      _valorStr = _formatoValido ? Utilidades::dblToStdStr(_valorDbl, 3) :
           valorStr;
10 }
11 ValorQuantitativo::ValorQuantitativo(const double valorDbl)
12 {
```

```
13      _valorStr = Utilidades::dblToStdStr(valorDbl, 3);
14      _valorDbl = valorDbl;
15      _formatoValido = !isnan(valorDbl);
16      _missing = (_valorStr == "") ? true : false;
17  }
18
19  double ValorQuantitativo::valorDbl() const
20  {
21      if (!_formatoValido)
22      {
23          return numeric_limits<double>::quiet_NaN();
24      }
25
26      return _valorDbl;
27  }
28
29  bool ValorQuantitativo::formatoValido() const
30  {
31      return _formatoValido;
32  }
33
34  bool ValorQuantitativo::operator ==(const ValorBase &outro) const
35  {
36      const ValorQuantitativo *const outraPtr = dynamic_cast<const
            ValorQuantitativo *const>(&outro);
37      return (outraPtr != NULL) && (_valorDbl == outraPtr->_valorDbl) &&
            _formatoValido == outraPtr->_formatoValido;
38  }
39
40  bool ValorQuantitativo::operator <(const ValorBase &outro) const
41  {
42      const ValorQuantitativo *const outraPtr = dynamic_cast<const
            ValorQuantitativo *const>(&outro);
43
44      if (outraPtr == NULL) //comparação entre valores de tipos diferentes,
            qualitativa precede quantitativa
45      {
46          return false;
47      }
48
49
50      if ((!this->formatoValido() || this->missing()) && (!outraPtr->
            formatoValido() || outraPtr->missing()))
51      {
52          return false;
53      }
54
55      if (!this->formatoValido() || this->missing())
56      {
57          return true;
58      }
59
60      if (!outraPtr->formatoValido() || outraPtr->missing())
61      {
62          return false;
63      }
64
65      return this->valorDbl() < outraPtr->valorDbl();
66  }
67
68  ValorQuantitativo *ValorQuantitativo::clone() const
69  {
70      return new ValorQuantitativo(*this);
71  }
```

Listagem B.131: valor_quantitativo.cpp

```
1  #ifndef VALORQUANTITATIVO_H
2  #define VALORQUANTITATIVO_H
3
4  #include "valor_base.h"
5
6  using namespace std;
7
8  class ValorQuantitativo : public ValorBase
9  {
10 public:
11     ValorQuantitativo(const string &valorStr);
12     ValorQuantitativo(const double valorDbl);
```

```
13      bool formatoValido() const;
14      bool operator <(const ValorBase &outro) const;
15      bool operator ==(const ValorBase &outro) const;
16      double valorDbl() const;
17      ValorQuantitativo *clone() const;
18  private:
19      double _valorDbl;
20      bool _formatoValido;
21  };
22  #endif
```

Listagem B.132: valor_quantitativo.h

B.3 SUBPROJETO LIB_FIT

```
1  ï»¿<?xml version="1.0" encoding="utf-8"?>
2  <Project DefaultTargets="Build" ToolsVersion="4.0" xmlns="http://schemas.
       microsoft.com/developer/msbuild/2003">
3    <ItemGroup Label="ProjectConfigurations">
4      <ProjectConfiguration Include="Debug|Win32">
5        <Configuration>Debug</Configuration>
6        <Platform>Win32</Platform>
7      </ProjectConfiguration>
8      <ProjectConfiguration Include="Release|Win32">
9        <Configuration>Release</Configuration>
10       <Platform>Win32</Platform>
11     </ProjectConfiguration>
12   </ItemGroup>
13   <PropertyGroup Label="Globals">
14     <ProjectGuid>{7CC8E7FD-D035-45E8-AD6A-2E1B5DD9C0DC}</ProjectGuid>
15     <Keyword>Qt4VSv1.0</Keyword>
16   </PropertyGroup>
17   <Import Project="$(VCTargetsPath)\Microsoft.Cpp.Default.props" />
18   <PropertyGroup Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'"
         Label="Configuration">
19     <ConfigurationType>DynamicLibrary</ConfigurationType>
20   </PropertyGroup>
21   <PropertyGroup Condition="'$(Configuration)|$(Platform)'=='Release|Win32'
       " Label="Configuration">
22     <ConfigurationType>DynamicLibrary</ConfigurationType>
23   </PropertyGroup>
24   <Import Project="$(VCTargetsPath)\Microsoft.Cpp.props" />
25   <ImportGroup Label="ExtensionSettings">
26   </ImportGroup>
27   <ImportGroup Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'"
         Label="PropertySheets">
28     <Import Project="$(UserRootDir)\Microsoft.Cpp.$(Platform).user.props"
           Condition="exists('$(UserRootDir)\Microsoft.Cpp.$(Platform).user.
           props')" Label="LocalAppDataPlatform" />
29   </ImportGroup>
30   <ImportGroup Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
         Label="PropertySheets">
31     <Import Project="$(UserRootDir)\Microsoft.Cpp.$(Platform).user.props"
           Condition="exists('$(UserRootDir)\Microsoft.Cpp.$(Platform).user.
           props')" Label="LocalAppDataPlatform" />
32   </ImportGroup>
33   <PropertyGroup Label="UserMacros" />
34   <PropertyGroup>
35     <_ProjectFileVersion>10.0.30319.1</_ProjectFileVersion>
36     <CodeAnalysisRuleSet Condition="'$(Configuration)|$(Platform)'=='Debug|
           Win32'">AllRules.ruleset</CodeAnalysisRuleSet>
37     <CodeAnalysisRules Condition="'$(Configuration)|$(Platform)'=='Debug|
           Win32'" />
38     <CodeAnalysisRuleAssemblies Condition="'$(Configuration)|$(Platform)
           '=='Debug|Win32'" />
39     <CodeAnalysisRuleSet Condition="'$(Configuration)|$(Platform)'=='
           Release|Win32'">AllRules.ruleset</CodeAnalysisRuleSet>
40     <CodeAnalysisRules Condition="'$(Configuration)|$(Platform)'=='Release|
           Win32'" />
41     <CodeAnalysisRuleAssemblies Condition="'$(Configuration)|$(Platform)
           '=='Release|Win32'" />
42     <OutDir Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">$(
           SolutionDir)$(Platform)\$(Configuration)\</OutDir>
```

```
43      <OutDir Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">$(
            SolutionDir)$(Platform)\$(Configuration)\</OutDir>
44    </PropertyGroup>
45    <ItemDefinitionGroup Condition="'$(Configuration)|$(Platform)'=='Debug|
          Win32'">
46      <ClCompile>
47        <PreprocessorDefinitions>UNICODE;WIN32;QT_DLL;LIB_FIT_LIB;%(
              PreprocessorDefinitions)</PreprocessorDefinitions>
48        <AdditionalIncludeDirectories>.;.\GeneratedFiles;.\GeneratedFiles\$(
              ConfigurationName);$(QTDIR)\include;.\includes;%(
              AdditionalIncludeDirectories)</AdditionalIncludeDirectories>
49        <Optimization>Disabled</Optimization>
50        <DebugInformationFormat>ProgramDatabase</DebugInformationFormat>
51        <RuntimeLibrary>MultiThreadedDebugDLL</RuntimeLibrary>
52        <TreatWChar_tAsBuiltInType>false</TreatWChar_tAsBuiltInType>
53      </ClCompile>
54      <Link>
55        <SubSystem>Windows</SubSystem>
56        <OutputFile>$(OutDir)\$(ProjectName).dll</OutputFile>
57        <AdditionalLibraryDirectories>$(QTDIR)\lib;%(
              AdditionalLibraryDirectories)</AdditionalLibraryDirectories>
58        <GenerateDebugInformation>true</GenerateDebugInformation>
59        <AdditionalDependencies>qtmaind.lib;%(AdditionalDependencies)</
              AdditionalDependencies>
60      </Link>
61    </ItemDefinitionGroup>
62    <ItemDefinitionGroup Condition="'$(Configuration)|$(Platform)'=='Release|
          Win32'">
63      <ClCompile>
64        <PreprocessorDefinitions>UNICODE;WIN32;QT_DLL;QT_NO_DEBUG;NDEBUG;
              LIB_FIT_LIB;%(PreprocessorDefinitions)</PreprocessorDefinitions>
65        <AdditionalIncludeDirectories>.;.\GeneratedFiles;.\GeneratedFiles\$(
              ConfigurationName);$(QTDIR)\include;.\includes;%(
              AdditionalIncludeDirectories)</AdditionalIncludeDirectories>
66        <DebugInformationFormat>
67        </DebugInformationFormat>
68        <RuntimeLibrary>MultiThreadedDLL</RuntimeLibrary>
69        <TreatWChar_tAsBuiltInType>false</TreatWChar_tAsBuiltInType>
70      </ClCompile>
71      <Link>
72        <SubSystem>Windows</SubSystem>
73        <OutputFile>$(OutDir)\$(ProjectName).dll</OutputFile>
74        <AdditionalLibraryDirectories>$(QTDIR)\lib;%(
              AdditionalLibraryDirectories)</AdditionalLibraryDirectories>
75        <GenerateDebugInformation>false</GenerateDebugInformation>
76        <AdditionalDependencies>qtmain.lib;%(AdditionalDependencies)</
              AdditionalDependencies>
77      </Link>
78    </ItemDefinitionGroup>
79    <ItemGroup>
80      <ClCompile Include="src\common\fit_utils.cpp" />
81      <ClCompile Include="src\common\freq_table_entry.cpp" />
82      <ClCompile Include="src\distributions\base_distrib.cpp" />
83      <ClCompile Include="src\distributions\beta_distrib.cpp" />
84      <ClCompile Include="src\distributions\expo_distrib.cpp" />
85      <ClCompile Include="src\distributions\gamm_distrib.cpp" />
86      <ClCompile Include="src\distributions\lognorm_distrib.cpp" />
87      <ClCompile Include="src\distributions\norm_distrib.cpp" />
88      <ClCompile Include="src\distributions\tria_distrib.cpp" />
89      <ClCompile Include="src\distributions\unif_distrib.cpp" />
90      <ClCompile Include="src\distributions\weib_distrib.cpp" />
91      <ClCompile Include="src\fit.cpp" />
92      <ClCompile Include="src\freq_table.cpp" />
93    </ItemGroup>
94    <ItemGroup>
95      <ClInclude Include="includes\common\fit_utils.hpp" />
96      <ClInclude Include="includes\common\freq_table_entry.h" />
97      <ClInclude Include="includes\distribution\base_distrib.h" />
98      <ClInclude Include="includes\distribution\beta_distrib.h" />
99      <ClInclude Include="includes\distribution\expo_distrib.h" />
100     <ClInclude Include="includes\distribution\gamm_distrib.h" />
101     <ClInclude Include="includes\distribution\lognorm_distrib.h" />
102     <ClInclude Include="includes\distribution\norm_distrib.h" />
103     <ClInclude Include="includes\distribution\tria_distrib.h" />
104     <ClInclude Include="includes\distribution\unif_distrib.h" />
105     <ClInclude Include="includes\distribution\weib_distrib.h" />
106     <ClInclude Include="includes\fit.h" />
107     <ClInclude Include="includes\fit_defs.h" />
108     <ClInclude Include="includes\freq_table.h" />
109   </ItemGroup>
110   <Import Project="$(VCTargetsPath)\Microsoft.Cpp.targets" />
111   <ImportGroup Label="ExtensionTargets">
```

```
112     </ImportGroup>
113     <ProjectExtensions>
114       <VisualStudio>
115         <UserProperties UicDir=".\GeneratedFiles" MocDir=".\GeneratedFiles\$(
                ConfigurationName)" MocOptions="" RccDir=".\GeneratedFiles"
                lupdateOnBuild="0" lupdateOptions="" lreleaseOptions=""
                Qt5Version_x0020_Win32="msvc2010_opengl" />
116       </VisualStudio>
117     </ProjectExtensions>
118   </Project>
```

Listagem B.133: lib_fit.vcxproj

```
1   ï»¿<?xml version="1.0" encoding="utf-8"?>
2   <Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/
        msbuild/2003">
3     <ItemGroup>
4       <Filter Include="Source Files">
5         <UniqueIdentifier>{4FC737F1-C7A5-4376-A066-2A32D752A2FF}</
              UniqueIdentifier>
6         <Extensions>cpp;cxx;c;def</Extensions>
7       </Filter>
8       <Filter Include="Header Files">
9         <UniqueIdentifier>{93995380-89BD-4b04-88EB-625FBE52EBFB}</
              UniqueIdentifier>
10        <Extensions>h</Extensions>
11      </Filter>
12      <Filter Include="Form Files">
13        <UniqueIdentifier>{99349809-55BA-4b9d-BF79-8FDBB0286EB3}</
              UniqueIdentifier>
14        <Extensions>ui</Extensions>
15      </Filter>
16      <Filter Include="Resource Files">
17        <UniqueIdentifier>{D9D6E242-F8AF-46E4-B9FD-80ECBC20BA3E}</
              UniqueIdentifier>
18        <Extensions>qrc;*</Extensions>
19        <ParseFiles>false</ParseFiles>
20      </Filter>
21      <Filter Include="Generated Files">
22        <UniqueIdentifier>{71ED8ED8-ACB9-4CE9-BBE1-E00B30144E11}</
              UniqueIdentifier>
23        <Extensions>moc;h;cpp</Extensions>
24        <SourceControlFiles>False</SourceControlFiles>
25      </Filter>
26      <Filter Include="Header Files\includes">
27        <UniqueIdentifier>{ec7f8ae2-d21e-4a8c-911a-1d15ab4cbf8c}</
              UniqueIdentifier>
28      </Filter>
29      <Filter Include="Source Files\src">
30        <UniqueIdentifier>{ca7451c9-2453-4ea0-ad47-a0184e8ba24d}</
              UniqueIdentifier>
31      </Filter>
32      <Filter Include="Source Files\src\common">
33        <UniqueIdentifier>{e25c0284-b2d6-42f6-bfce-0d7dd6050a02}</
              UniqueIdentifier>
34      </Filter>
35      <Filter Include="Header Files\includes\common">
36        <UniqueIdentifier>{ed4f5fb7-1361-4f6a-99db-503564512202}</
              UniqueIdentifier>
37      </Filter>
38      <Filter Include="Header Files\includes\distribution">
39        <UniqueIdentifier>{820432ba-8190-419d-abf0-09b99d63c440}</
              UniqueIdentifier>
40      </Filter>
41      <Filter Include="Source Files\src\distribution">
42        <UniqueIdentifier>{d0b9c637-f26b-4b21-9544-332a215f496c}</
              UniqueIdentifier>
43      </Filter>
44    </ItemGroup>
45    <ItemGroup>
46      <ClCompile Include="src\common\fit_utils.cpp">
47        <Filter>Source Files\src\common</Filter>
48      </ClCompile>
49      <ClCompile Include="src\common\freq_table_entry.cpp">
50        <Filter>Source Files\src\common</Filter>
51      </ClCompile>
52      <ClCompile Include="src\distributions\base_distrib.cpp">
53        <Filter>Source Files\src\distribution</Filter>
54      </ClCompile>
```

```
55      <ClCompile Include="src\distributions\beta_distrib.cpp">
56        <Filter>Source Files\src\distribution</Filter>
57      </ClCompile>
58      <ClCompile Include="src\distributions\expo_distrib.cpp">
59        <Filter>Source Files\src\distribution</Filter>
60      </ClCompile>
61      <ClCompile Include="src\distributions\gamm_distrib.cpp">
62        <Filter>Source Files\src\distribution</Filter>
63      </ClCompile>
64      <ClCompile Include="src\distributions\lognorm_distrib.cpp">
65        <Filter>Source Files\src\distribution</Filter>
66      </ClCompile>
67      <ClCompile Include="src\distributions\norm_distrib.cpp">
68        <Filter>Source Files\src\distribution</Filter>
69      </ClCompile>
70      <ClCompile Include="src\distributions\tria_distrib.cpp">
71        <Filter>Source Files\src\distribution</Filter>
72      </ClCompile>
73      <ClCompile Include="src\distributions\unif_distrib.cpp">
74        <Filter>Source Files\src\distribution</Filter>
75      </ClCompile>
76      <ClCompile Include="src\distributions\weib_distrib.cpp">
77        <Filter>Source Files\src\distribution</Filter>
78      </ClCompile>
79      <ClCompile Include="src\freq_table.cpp">
80        <Filter>Source Files\src</Filter>
81      </ClCompile>
82      <ClCompile Include="src\fit.cpp">
83        <Filter>Source Files\src</Filter>
84      </ClCompile>
85    </ItemGroup>
86    <ItemGroup>
87      <ClInclude Include="includes\common\fit_utils.hpp">
88        <Filter>Header Files\includes\common</Filter>
89      </ClInclude>
90      <ClInclude Include="includes\common\freq_table_entry.h">
91        <Filter>Header Files\includes\common</Filter>
92      </ClInclude>
93      <ClInclude Include="includes\distribution\base_distrib.h">
94        <Filter>Header Files\includes\distribution</Filter>
95      </ClInclude>
96      <ClInclude Include="includes\distribution\beta_distrib.h">
97        <Filter>Header Files\includes\distribution</Filter>
98      </ClInclude>
99      <ClInclude Include="includes\distribution\expo_distrib.h">
100       <Filter>Header Files\includes\distribution</Filter>
101     </ClInclude>
102     <ClInclude Include="includes\distribution\gamm_distrib.h">
103       <Filter>Header Files\includes\distribution</Filter>
104     </ClInclude>
105     <ClInclude Include="includes\distribution\lognorm_distrib.h">
106       <Filter>Header Files\includes\distribution</Filter>
107     </ClInclude>
108     <ClInclude Include="includes\distribution\norm_distrib.h">
109       <Filter>Header Files\includes\distribution</Filter>
110     </ClInclude>
111     <ClInclude Include="includes\distribution\tria_distrib.h">
112       <Filter>Header Files\includes\distribution</Filter>
113     </ClInclude>
114     <ClInclude Include="includes\distribution\unif_distrib.h">
115       <Filter>Header Files\includes\distribution</Filter>
116     </ClInclude>
117     <ClInclude Include="includes\distribution\weib_distrib.h">
118       <Filter>Header Files\includes\distribution</Filter>
119     </ClInclude>
120     <ClInclude Include="includes\fit.h">
121       <Filter>Header Files\includes</Filter>
122     </ClInclude>
123     <ClInclude Include="includes\fit_defs.h">
124       <Filter>Header Files\includes</Filter>
125     </ClInclude>
126     <ClInclude Include="includes\freq_table.h">
127       <Filter>Header Files\includes</Filter>
128     </ClInclude>
129   </ItemGroup>
130 </Project>
```

Listagem B.134: lib_fit.vcxproj.filters

```
1  //
   ------------------------------------------------------------------------
2  //    'FIT', a library for fitting statistical distribution
3  //    Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //    This library is free software; you can redistribute it and/or
6  //    modify it under the terms of the GNU Lesser General Public
7  //    License as published by the Free Software Foundation; either
8  //    version 2.1 of the License, or (at your option) any later version.
9  //
10 //    This library is distributed in the hope that it will be useful,
11 //    but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //    Lesser General Public License for more details.
14 //
15 //    You should have received a copy of the GNU Lesser General Public
16 //    License along with this library; if not, write to the Free Software
17 //    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
   //    USA
18 //
   ------------------------------------------------------------------------

19
20 #include <fit.h>
21 #include <common/fit_utils.hpp>
22
23 fit::distrib_ptr fit::get_distribution(const data_set& values, int type)
24 {
25     base_distrib* distrib_pointer = 0;
26
27     switch (type)
28     {
29         case BETA:
30             distrib_pointer = new beta_distrib(values);
31             break;
32         case EXPONENTIAL:
33             distrib_pointer = new expo_distrib(values);
34             break;
35         case GAMMA:
36             distrib_pointer = new gamm_distrib(values);
37             break;
38         case LOGNORMAL:
39             distrib_pointer = new lognorm_distrib(values);
40             break;
41         case NORMAL:
42             distrib_pointer = new norm_distrib(values);
43             break;
44         case TRIANGULAR:
45             distrib_pointer = new tria_distrib(values);
46             break;
47         case UNIFORM:
48             distrib_pointer = new unif_distrib(values);
49             break;
50         case WEIBULL:
51             distrib_pointer = new weib_distrib(values);
52             break;
53         default:
54             throw distribution_exception("Distribution not found");
55             break;
56     }
57
58     return distrib_ptr(distrib_pointer);
59 }
60
61 fit::distrib_ptr fit::get_distribution(const params_list& params, int type)
62 {
63     base_distrib* distrib_pointer = 0;
64
65     switch (type)
66     {
67         case BETA:
68             distrib_pointer = new beta_distrib(params);
69             break;
70         case EXPONENTIAL:
71             distrib_pointer = new expo_distrib(params);
72             break;
73         case GAMMA:
74             distrib_pointer = new gamm_distrib(params);
75             break;
76         case LOGNORMAL:
77             distrib_pointer = new lognorm_distrib(params);
```

```
78              break;
79          case NORMAL:
80              distrib_pointer = new norm_distrib(params);
81              break;
82          case TRIANGULAR:
83              distrib_pointer = new tria_distrib(params);
84              break;
85          case UNIFORM:
86              distrib_pointer = new unif_distrib(params);
87              break;
88          case WEIBULL:
89              distrib_pointer = new weib_distrib(params);
90              break;
91          default:
92              throw distribution_exception("Distribution not found");
93              break;
94      }
95
96      return distrib_ptr(distrib_pointer);
97  }
98
99  fit::distrib_ptr fit::get_default_distribution(int type)
100 {
101     base_distrib* distrib_pointer = 0;
102
103     params_list params = params_list();
104     params.push_back(1);
105
106     switch (type)
107     {
108         case BETA:
109             params.push_back(2);
110             distrib_pointer = new beta_distrib(params);
111             break;
112         case EXPONENTIAL:
113             distrib_pointer = new expo_distrib(params);
114             break;
115         case GAMMA:
116             params.push_back(2);
117             distrib_pointer = new gamm_distrib(params);
118             break;
119         case LOGNORMAL:
120             distrib_pointer = new lognorm_distrib(params);
121             break;
122         case NORMAL:
123             params.push_back(2);
124             distrib_pointer = new norm_distrib(params);
125             break;
126         case TRIANGULAR:
127             params.push_back(2);
128             params.push_back(3);
129             distrib_pointer = new tria_distrib(params);
130             break;
131         case UNIFORM:
132             params.push_back(2);
133             distrib_pointer = new unif_distrib(params);
134             break;
135         case WEIBULL:
136             params.push_back(2);
137             distrib_pointer = new weib_distrib(params);
138             break;
139         default:
140             throw distribution_exception("Distribution not found");
141             break;
142     }
143
144     return distrib_ptr(distrib_pointer);
145 }
146
147 fit::freq_table fit::get_freq_table(const data_set& values, int dist_type,
        int criteria, unsigned int max_bins)
148 {
149     distrib_ptr dist = get_distribution(values, dist_type);
150     return freq_table(values, *dist, (_bin_criteria) criteria, max_bins);
151 }
152
153 fit::freq_table fit::get_freq_table(const data_set& values, const
        base_distrib& distrib, int criteria, unsigned int max_bins)
154 {
155     return freq_table(values, distrib, (_bin_criteria) criteria, max_bins);
156 }
157
```

```
158 fit::chisquare_test_result fit::get_chisquare_test(const freq_table& table)
159 {
160     chisquare_test_result result = {0, 0, 0};
161
162     result.chi_statistic = utils::_chisquare_statistic(table);
163     result.degrees_freedom = table.get_df();
164     result.chi_p_value = utils::_chisquare_pf(result.chi_statistic, result.
          degrees_freedom);
165
166     return result;
167 }
```

Listagem B.135: fit.cpp

```
1  //
   ------------------------------------------------------------------------
2  //    'FIT', a library for fitting statistical distribution
3  //    Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //    This library is free software; you can redistribute it and/or
6  //    modify it under the terms of the GNU Lesser General Public
7  //    License as published by the Free Software Foundation; either
8  //    version 2.1 of the License, or (at your option) any later version.
9  //
10 //    This library is distributed in the hope that it will be useful,
11 //    but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //    Lesser General Public License for more details.
14 //
15 //    You should have received a copy of the GNU Lesser General Public
16 //    License along with this library; if not, write to the Free Software
17 //    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
   ------------------------------------------------------------------------

19
20 #ifndef FIT_H_
21 #define FIT_H_
22
23 #include <memory>
24
25 #include "distribution/beta_distrib.h"
26 #include "distribution/expo_distrib.h"
27 #include "distribution/lognorm_distrib.h"
28 #include "distribution/tria_distrib.h"
29 #include "distribution/unif_distrib.h"
30 #include "distribution/weib_distrib.h"
31
32 #include "freq_table.h"
33
34 #if __GNUC__ >= 4
35 #pragma GCC visibility push(default)
36 #endif //__GNUC__
37
38 namespace fit
39 {
40     DLLEXPORT enum _distribution_type
41     {
42         BETA,
43         EXPONENTIAL,
44         GAMMA,
45         LOGNORMAL,
46         NORMAL,
47         TRIANGULAR,
48         UNIFORM,
49         WEIBULL
50     };
51
52     DLLEXPORT struct chisquare_test_result
53     {
54         data_type chi_statistic; //the test statistic
55         unsigned int degrees_freedom; //the number of degrees of freedom
56
57         /* The significance probability of the test.
58          * A small value of chi_p_value indicates a
59          * significant difference between the data set
60          * and the choosen distribution.
```

```
61          */
62         data_type chi_p_value;
63     };
64
65     typedef std::auto_ptr<base_distrib> distrib_ptr;
66
67     DLLEXPORT distrib_ptr get_distribution(const data_set& values, int
           dist_type);
68     DLLEXPORT distrib_ptr get_distribution(const params_list& params, int
           dist_type);
69     DLLEXPORT distrib_ptr get_default_distribution(int dist_type);
70
71     DLLEXPORT freq_table get_freq_table(const data_set& values, int
           dist_type, int criteria, unsigned int max_bins = _MAX_BINS);
72     DLLEXPORT freq_table get_freq_table(const data_set& values, const
           base_distrib& distrib, int criteria, unsigned int max_bins =
           _MAX_BINS);
73
74     DLLEXPORT chisquare_test_result get_chisquare_test(const freq_table&
           table);
75
76 #if __GNUC__ >= 4
77 #pragma GCC visibility pop
78 #endif //__GNUC__
79
80 }
81
82 #endif /*FIT_H_*/
```

Listagem B.136: fit.h

```
1  //
   ------------------------------------------------------------------------
2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
   ------------------------------------------------------------------------

19
20 #ifndef FIT_DEFS_H_
21 #define FIT_DEFS_H_
22
23 #include <limits>
24 #include <list>
25 #include <string>
26
27 #ifdef _MSC_VER
28     #define DLLEXPORT __declspec(dllexport)
29 #else
30     #define DLLEXPORT
31 #endif
32
33 namespace fit
34 {
35     /* Defines the precision */
36     typedef double                      data_type;
37     typedef std::list<data_type>        data_set;
38
39     /* Frequency table constraint */
40     enum _bin_criteria
41     {
42         STURGES,
43         SQUAREROOT,
```

```
44          OTHER
45      };
46      const unsigned int _MAX_BINS = 40;
47
48      /* Exceptions */
49      DLLEXPORT struct value_exception
50      {
51          std::string msg;
52          value_exception(std::string _msg) : msg(_msg) {}
53      };
54
55      DLLEXPORT struct freq_table_exception
56      {
57          std::string msg;
58          freq_table_exception(std::string _msg) : msg(_msg) {}
59      };
60
61      DLLEXPORT struct distribution_exception
62      {
63          std::string msg;
64          distribution_exception(std::string _msg) : msg(_msg) {}
65      };
66  };
67
68  #endif /*FIT_DEFS_H_*/
```

Listagem B.137: fit_defs.h

```
1   //
2   //     'FIT', a library for fitting statistical distribution
3   //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4   //
5   //     This library is free software; you can redistribute it and/or
6   //     modify it under the terms of the GNU Lesser General Public
7   //     License as published by the Free Software Foundation; either
8   //     version 2.1 of the License, or (at your option) any later version.
9   //
10  //     This library is distributed in the hope that it will be useful,
11  //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12  //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13  //     Lesser General Public License for more details.
14  //
15  //     You should have received a copy of the GNU Lesser General Public
16  //     License along with this library; if not, write to the Free Software
17  //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
        USA
18  //
19
20  #include <common/fit_utils.hpp>
21  #include <cmath>
22
23  using namespace fit;
24
25  unsigned int fit::utils::_bin_width(_bin_criteria _criteria, int
        _samples_number, unsigned int _max_bins) {
26      if (_samples_number <= 0)
27          return 0;
28
29      int ret = 0;
30      switch (_criteria) {
31      case SQUAREROOT:
32          ret = std::floor (std::sqrt((double)_samples_number) + 0.5);
33          break;
34      case STURGES:
35          ret = std::floor (1.5 + 3.222 * std::log10((double)_samples_number)
                );
36          break;
37      case OTHER:
38      default:
39          return _max_bins;
40      }
41
42      return ret < _max_bins ? ret : _max_bins;
43  }
44
```

```
45 data_type fit::utils::_chisquare_statistic(const freq_table& _table)
46 {
47     if (_table.size() <= 5)
48         throw freq_table_exception("Chi-Square test is not applicable on
               this frequency table. Number of intervals <= 5");
49
50     data_type obs_freq, exp_freq;
51     data_type result = 0.0;
52     for (freq_table_t::const_iterator i = _table.begin(); i != _table.end()
           ; i++)
53     {
54         obs_freq = i->get_observed_freq();
55         exp_freq = i->get_expected_freq();
56         result += std::pow(obs_freq - exp_freq, 2) / exp_freq;
57     }
58     return result;
59 }
60
61 data_type fit::utils::_chisquare_pf(data_type chisquare_stat, unsigned int
       df)
62 {
63     return gammq(df / 2.0, chisquare_stat / 2.0);
64 }
65
66 data_type fit::utils::gammln(data_type xx)
67 //Returns de value of ln[Î“(xx)] for xx > 0.
68 {
69     //
       ------------------------------------------------------------------------

70     /*  Internal arithmetic will be done in double precision, a nicety that
           you can omit if five figure
71      accuracy is good enough. */
72
73     data_type x, y, tmp, ser;
74     data_type cof[6] = { 76.18009172947146, -86.50532032941677,
75             24.01409824083091, -1.231739572450155, 0.1208650973866179e-2,
76             -0.5395239384953e-5 };
77     int j;
78     y = x = xx;
79     tmp = x + 5.5;
80     tmp -= (x + 0.5) * log (tmp);
81     ser = 1.000000000190015;
82     for (j = 0; j <= 5; j++)
83         ser += cof[j] / ++y;
84     return -tmp + log (2.5066282746310005 * ser / x);
85 }
86
87 void fit::utils::gcf(data_type *gammcf, data_type a, data_type x,
88         data_type *gln)
89 /*Returns the incomplete gamma function Q(a, x) evaluated by its continued
       fraction represen-
90  tation as gammcf. Also returns ln Î“(a) as gln.*/
91 {
92     int i;
93     data_type an, b, c, d, del, h;
94     *gln = gammln (a);
95     b = x + 1.0- a;
96     c = 1.0 / DATAMIN;
97     d = 1.0 / b;
98     h = d;
99     for (i = 1; i <= ITMAX; i++) {
100        an = -i * (i - a);
101        b += 2.0;
102        d = an * d + b;
103        if (fabs (d) < DATAMIN)
104            d = DATAMIN;
105        c = b + an / c;
106        if (fabs (c) < DATAMIN)
107            c = DATAMIN;
108        d = 1.0 / d;
109        del = d * c;
110        h *= del;
111        if (fabs (del - 1.0) < EPS)
112            break;
113    }
114    if (i > ITMAX)
115        throw value_exception("a too large, ITMAX too small in gcf");
116    *gammcf = exp (-x + a * log (x) - (*gln)) * h;
117 }
118
119 void fit::utils::gser(data_type *gamser, data_type a, data_type x,
```

```
120         data_type *gln) {
121     /*Returns the incomplete gamma function P (a, x) evaluated by its
            series representation as gamser.
122      Also returns ln Î“(a) as gln.*/
123
124     int n;
125     data_type sum, del, ap;
126     *gln = gammln (a);
127     if (x <= 0.0) {
128         if (x < 0.0)
129             throw value_exception("x less than 0 in routine gser");
130         *gamser = 0.0;
131         return;
132     } else {
133         ap = a;
134         del = sum = 1.0 / a;
135         for (n = 1; n <= ITMAX; n++) {
136             ++ap;
137             del *= x / ap;
138             sum += del;
139             if (fabs (del) < fabs (sum) * EPS) {
140                 *gamser =sum * exp (-x + a * log (x) - (*gln));
141                 return;
142             }
143         }
144         throw value_exception("a too large, ITMAX too small in routine gser
                ");
145         return;
146     }
147 }
148
149 data_type fit::utils::gammp(data_type a, data_type x)
150 //Returns the incomplete gamma function P (a, x).
151 {
152     data_type gamser, gammcf, gln;
153     if (x < 0.0|| a <= 0.0)
154         throw value_exception("Invalid arguments in routine gammp");
155     if (x < (a+1.0)) {
156         gser(&gamser, a, x, &gln);
157         return gamser;
158     } else {
159         gcf(&gammcf, a, x, &gln);
160         return 1.0-gammcf;
161     }
162 }
163
164 data_type fit::utils::gammq(data_type a, data_type x)
165 //Returns the incomplete gamma function Q(a, x) = 1 â^' P (a, x).
166 {
167     data_type gamser, gammcf, gln;
168     if (x < 0.0|| a <= 0.0)
169         throw value_exception("Invalid arguments in routine gammq");
170     if (x < (a + 1.0)) {
171         gser (&gamser, a, x, &gln);
172         return 1.0 - gamser;
173     } else {
174         gcf (&gammcf, a, x, &gln);
175         return gammcf;
176     }
177 }
178
179 //=====================================================================
180
181 data_type fit::utils::betai(data_type a, data_type b, data_type x)
182 //Returns the incomplete beta function Ix (a, b).
183 {
184     data_type bt;
185     if (x < 0.0 || x > 1.0) throw value_exception("Bad x in routine betai:
            ");
186
187     if (x == 0.0 || x == 1.0)
188     bt=0.0;
189     else
190     bt=exp(gammln(a+b)-gammln(a)-gammln(b)+a*log(x)+b*log(1.0-x));
191     if (x < (a+1.0)/(a+b+2.0))
192     return bt*betacf(a,b,x)/a;
193     else
194     return 1.0-bt*betacf(b,a,1.0-x)/b;
195 }
196
197 data_type fit::utils::betacf(data_type a, data_type b, data_type x) {
198     int m, m2;
```

```
199     float aa, c, d, del, h, qab, qam, qap;
200     qab=a+b;
201     qap=a+1.0;
202     qam=a-1.0;
203     c=1.0;
204     d=1.0-qab*x/qap;
205     if (fabs(d) < DATAMIN)
206         d=DATAMIN;
207     d=1.0/d;
208     h=d;
209     for (m=1; m<=ITMAX; m++) {
210         m2=2*m;
211         aa=m*(b-m)*x/((qam+m2)*(a+m2));
212         d=1.0+aa*d;
213         if (fabs(d) < DATAMIN)
214             d=DATAMIN;
215         c=1.0+aa/c;
216         if (fabs(c) < DATAMIN)
217             c=DATAMIN;
218         d=1.0/d;
219         h *= d*c;
220         aa = -(a+m)*(qab+m)*x/((a+m2)*(qap+m2));
221         d=1.0+aa*d;
222         if (fabs(d) < DATAMIN)
223             d=DATAMIN;
224         c=1.0+aa/c;
225         if (fabs(c) < DATAMIN)
226             c=DATAMIN;
227         d=1.0/d;
228         del=d*c;
229         h *= del;
230
231         if (fabs(del-1.0) < EPS)
232             break;
233     }
234     if (m > ITMAX)
235         throw value_exception("a or b too big, or MAXIT too small in betacf
                ");
236
237     return h;
238 }
```

Listagem B.138: fit_utils.cpp

```
1  //
   ------------------------------------------------------------------------

2  //    'FIT', a library for fitting statistical distribution
3  //    Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //    This library is free software; you can redistribute it and/or
6  //    modify it under the terms of the GNU Lesser General Public
7  //    License as published by the Free Software Foundation; either
8  //    version 2.1 of the License, or (at your option) any later version.
9  //
10 //    This library is distributed in the hope that it will be useful,
11 //    but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //    Lesser General Public License for more details.
14 //
15 //    You should have received a copy of the GNU Lesser General Public
16 //    License along with this library; if not, write to the Free Software
17 //    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
   ------------------------------------------------------------------------

19
20 #ifndef FIT_UTILS_HPP_
21 #define FIT_UTILS_HPP_
22
23 #include "fit_defs.h"
24 #include "freq_table.h"
25
26 #include <cmath>
27 #include <list>
28
29 namespace fit
30 {
```

```
31      namespace utils
32      {
33          const int ITMAX = 100;
34          const float EPS = 3.0e-7;
35
36  #ifndef BORLAND
37          const data_type DATAMIN = std::numeric_limits<data_type>::
                denorm_min();
38  #else
39          const data_type DATAMIN = 1.0e-30; // It's approx. float min. No
                problem.
40  #endif
41
42          const long double PI = 3.14159265358979323846264338327950288419 68;
43
44          extern "C"
45          {
46          /* Gamma utils. From Numerical Recipes in C, 1992. */
47          data_type gammln(data_type xx);
48          void gcf(data_type *gammcf, data_type a, data_type x, data_type *
                gln);
49          void gser(data_type *gamser, data_type a, data_type x, data_type *
                gln);
50          data_type gammp(data_type a, data_type x);
51          data_type gammq(data_type a, data_type x);
52
53          /* Beta utils. From Numerical Recipes in C, 1992. */
54          data_type betacf(data_type a, data_type b, data_type x);
55          data_type betai(data_type a, data_type b, data_type x);
56          }
57
58          /*
59           * SQUAREROOT is the square root os _samples_number
60           * STURGES is 1.5 + 3.222 * log10(_samples_number)
61           * The maximum number os bins is _max_bin.
62           */
63          unsigned int _bin_width (_bin_criteria _criteria, int
                _samples_number, unsigned int _max_bins = _MAX_BINS);
64
65          data_type _chisquare_statistic(const freq_table& _table);
66          data_type _chisquare_pf(data_type chisquare_stat, unsigned int df);
                //chi-square probability function. Is an incomplete gamma
                function.
67
68          /* Statistical utilities */
69          template < class T > T _mean (const typename std::list < T >&
                values)
70          {
71              if (values.empty ())
72                  return 0;
73              T sum = 0;
74              for (typename std::list < T >::const_iterator i = values.begin
                    ();
75                   i != values.end (); i++)
76              {
77                  sum += *i;
78              }
79              return sum / values.size ();
80          }
81
82          template < class T > T _variance (const std::list < T >& values)
83          {
84              if (values.empty ())
85                  return 0;
86              T temp = 0;
87              T mean_value = mean (values);
88              for (typename std::list < T >::const_iterator i = values.begin
                    ();
89                   i != values.end (); i++)
90              {
91                  temp += (*i - mean_value) * (*i - mean_value);
92              }
93              return temp / (values.size () - 1);
94          }
95
96          template < class T > T _variance (T mean, const std::list < T >&
                values)
97          {
98              T temp = 0;
99              for (typename std::list < T >::const_iterator i = values.begin
                    ();
100                  i != values.end (); i++)
```

```
101             {
102                 temp += (*i - mean) * (*i - mean);
103             }
104             return temp / (values.size () - 1);
105         }
106
107         template < class T > T _std_dev (const std::list < T >& values)
108         {
109             if (values.empty ())
110                 return 0;
111             T var_value = _variance (values);
112             return std::sqrt (var_value);
113         }
114     };
115 };
116
117 #endif /*FIT_UTILS_HPP_*/
```

Listagem B.139: fit_utils.hpp

```
1  //
       ------------------------------------------------------------------------
2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
       ------------------------------------------------------------------------

19
20 #include <freq_table.h>
21 #include <common/fit_utils.hpp>
22 #include <cmath>
23 #include <algorithm>
24
25 using namespace fit;
26
27 freq_table::freq_table(const data_set& values, const base_distrib& distrib,
       _bin_criteria criteria, unsigned int max_bins)
28     : _values(values)
29 {
30     init_members(distrib, utils::_bin_width(criteria, _values.size(),
           max_bins));
31     optimize(distrib);
32     df = size() - distrib.get_params_number() - 1;
33 }
34
35 freq_table::freq_table(const data_set& values, const base_distrib& distrib,
       unsigned int intervals)
36     : _values(values)
37 {
38     init_members(distrib, intervals);
39     optimize(distrib);
40     df = size() - distrib.get_params_number() - 1;
41 }
42
43 void freq_table::init_members(const base_distrib& distrib, unsigned int
       intervals)
44 {
45     if (_values.empty())
46         throw freq_table_exception("No values on frequency table
               construction");
47
48     if (intervals <= 0)
49         throw freq_table_exception("Negative number of intervals on
               frequency table construction :" + intervals);
```

```
50
51      data_type minor = *min_element(_values.begin(), _values.end());
52      data_type major = *max_element(_values.begin(), _values.end());
53
54      data_type difference = major - minor;
55
56      if (difference != difference) //std::isnan(difference)
57          throw freq_table_exception("Constant values on frequency table
                construction");
58
59      data_type step = (difference * 1.01) / intervals; //Little work-around
            to work ok with float-point numbers.
60
61      /*
62       * intervals_values is a list where intervals correspondent
63       * values are classified.
64       */
65      std::vector<data_set> intervals_values(intervals);
66
67      /* classifies the values */
68      for (data_set::const_iterator i = _values.begin (); i != _values.end ()
            ; i++)
69      {
70          int auxtmp = std::floor( (*i - minor) / step );
71          intervals_values.at(auxtmp).push_back(*i);
72      }
73
74      for (unsigned int i = 0; i < intervals; i++)
75      {
76          data_type interv_min = minor + (i * step);
77          data_type interv_max = minor + ((i * step) + step);
78          data_type interv_exp_freq = distrib.expected_freq(interv_min,
                interv_max, _values.size());
79
80          freq_table_entry new_entry(intervals_values.at(i), interv_min,
                interv_max, interv_exp_freq);
81
82          push_back(new_entry);
83      }
84  }
85
86  void freq_table::optimize(const base_distrib& distrib)
87  {
88      divide_intervals(distrib);
89      merge_intervals();
90  }
91
92  void freq_table::divide_intervals(const base_distrib& distrib)
93  {
94      for (freq_table_t::iterator it = begin(); it != end() ; )
95      {
96          if ( it->get_observed_freq() > (10.0 * (_values.size()/size())) )
                //Barbetta criteria.
97          {
98              data_type minor = it->get_min_value();
99              data_type major = it->get_max_value();
100             data_type medium = (minor + major) / 2.0;
101             data_set actual_interval_v = it->get_entry_values();
102             data_set previous_interval_v;
103             data_set subsequent_interval_v;
104             for (data_set::const_iterator itV = actual_interval_v.begin();
                    itV != actual_interval_v.end(); itV++)
105             {
106                 if (*itV < medium)
107                 {
108                     previous_interval_v.push_back(*itV);
109                 }
110                 else
111                 {
112                     subsequent_interval_v.push_back(*itV);
113                 }
114             }
115             if (previous_interval_v.size() == 0)
116             //this happens when there are too many values repeated.
117             {
118                 it++;
119             }
120             else
121             {
122                 freq_table_entry previous_interval(previous_interval_v,
                        minor, medium, distrib.expected_freq(minor, medium,
                        _values.size()));
```

```
123                     freq_table_entry subsequent_interval(subsequent_interval_v,
                              medium, major, distrib.expected_freq(medium, major,
                            _values.size()));
124                     it = erase(it);
125                     it = insert(it,subsequent_interval);
126                     it = insert(it,previous_interval);
127                 }
128             }
129             else { it++; }
130         }
131 }
132
133 void freq_table::merge_intervals()
134 {
135     freq_table_t::iterator it_temp;
136     freq_table_t::iterator end_it = end();
137     end_it--;
138
139     data_type expected_freq_sum = 0;
140     for (freq_table_t::iterator it = begin(); it != end(); it++)
141     {
142         expected_freq_sum += it->get_expected_freq();
143     }
144
145     if (expected_freq_sum < 5) return; //return if it is impossible to
            merge.
146
147     //Merge entry whose expected frequency is less than 5
148     for (freq_table_t::iterator it = begin() ; it != end() ; )
149     {
150         if (it->get_expected_freq() < 5)
151         {
152             if (it == end_it)
153             {
154                 it_temp = it;
155                 it_temp--;
156                 it->merge(*it_temp);
157                 erase(it_temp);
158             }
159             else
160             {
161                 it_temp = it;
162                 it_temp++;
163                 it_temp->merge(*it);
164                 it = erase(it);
165             }
166         }
167         else
168         {
169             it++;
170         }
171     }
172 }
173
174 data_type freq_table::get_square_error() const
175 {
176     data_type sum = 0;
177     for (freq_table_t::const_iterator it = begin(); it != end(); it++)
178     {
179         data_type expfreq = it->get_expected_freq();
180         unsigned int obsfreq = it->get_observed_freq();
181         sum += (expfreq - obsfreq) * (expfreq - obsfreq);
182     }
183
184     return std::sqrt(sum) / size();
185 }
```

Listagem B.140: freq_table.cpp

```
1 //
  ------------------------------------------------------------------------

2 //     'FIT', a library for fitting statistical distribution
3 //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4 //
5 //     This library is free software; you can redistribute it and/or
6 //     modify it under the terms of the GNU Lesser General Public
7 //     License as published by the Free Software Foundation; either
8 //     version 2.1 of the License, or (at your option) any later version.
```

```
9  //
10 //    This library is distributed in the hope that it will be useful,
11 //    but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //    Lesser General Public License for more details.
14 //
15 //    You should have received a copy of the GNU Lesser General Public
16 //    License along with this library; if not, write to the Free Software
17 //    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
```

```
19
20 #ifndef FREQ_TABLE_H_
21 #define FREQ_TABLE_H_
22
23 #include "common/freq_table_entry.h"
24 #include "distribution/base_distrib.h"
25
26 namespace fit
27 {
28 typedef std::list<freq_table_entry> freq_table_t; //frequency table type
29 class freq_table : public freq_table_t
30 {
31 private:
32     data_set _values;
33     unsigned int df; //degrees of freedom
34
35     void init_members(const base_distrib& distrib, unsigned int intervals);
36     void divide_intervals(const base_distrib& distrib);
37     void merge_intervals();
38
39 protected:
40     /*
41      * This procedure divide information excess intervals
42      * and merge whoses have expected frequency less than 5.
43      * (See Chi-Square test requisits.)
44      */
45     void optimize(const base_distrib& distrib);
46
47 public:
48     freq_table(const data_set& values, const base_distrib& distrib,
           _bin_criteria criteria, unsigned int max_bins = _MAX_BINS);
49     freq_table(const data_set& values, const base_distrib& distrib,
           unsigned int intervals);
50     ~freq_table() {};
51
52     data_type get_square_error() const;
53     unsigned int get_df() const { return df; }
54 };
55 }
56
57 #endif /*FREQ_TABLE_H_*/
```

Listagem B.141: freq_table.h

```
1  //
```

```
2  //    'FIT', a library for fitting statistical distribution
3  //    Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //    This library is free software; you can redistribute it and/or
6  //    modify it under the terms of the GNU Lesser General Public
7  //    License as published by the Free Software Foundation; either
8  //    version 2.1 of the License, or (at your option) any later version.
9  //
10 //    This library is distributed in the hope that it will be useful,
11 //    but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //    Lesser General Public License for more details.
14 //
15 //    You should have received a copy of the GNU Lesser General Public
16 //    License along with this library; if not, write to the Free Software
17 //    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
```

```
19
20 #include <common/freq_table_entry.h>
21
22 using namespace fit;
23
24 freq_table_entry::freq_table_entry(const data_set& entry_values, data_type
       min_value,
25         data_type max_value, data_type expected_freq)
26 : _entry_values(entry_values), _expected_freq(expected_freq),
27   _min_value(min_value), _max_value(max_value)
28 {
29 }
30
31 void freq_table_entry::set_expected_freq(data_type expected_freq)
32 {
33     _expected_freq = expected_freq < 0 ? 0 : expected_freq;
34 }
35
36 data_type freq_table_entry::get_min_value() const
37 {
38     return _min_value;
39 }
40
41 data_type freq_table_entry::get_max_value() const
42 {
43     return _max_value;
44 }
45
46 unsigned int freq_table_entry::get_observed_freq() const
47 {
48     return _entry_values.size();
49 }
50
51 data_type freq_table_entry::get_expected_freq() const
52 {
53     return _expected_freq;
54 }
55
56 data_set freq_table_entry::get_entry_values() const
57 {
58     return _entry_values;
59 }
60
61 void freq_table_entry::merge(const freq_table_entry& other_entry)
62 {
63     data_set tmpset = other_entry.get_entry_values();
64     _entry_values.merge(tmpset);
65
66     _expected_freq += other_entry.get_expected_freq();
67 }
```

Listagem B.142: freq_table_entry.cpp

```
1 //
    ------------------------------------------------------------------------

2 //     'FIT', a library for fitting statistical distribution
3 //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4 //
5 //     This library is free software; you can redistribute it and/or
6 //     modify it under the terms of the GNU Lesser General Public
7 //     License as published by the Free Software Foundation; either
8 //     version 2.1 of the License, or (at your option) any later version.
9 //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
    ------------------------------------------------------------------------

19
20 #ifndef FREQ_TABLE_ENTRY_H_
21 #define FREQ_TABLE_ENTRY_H_
```

```
22
23 #include "fit_defs.h"
24
25 namespace fit
26 {
27 class freq_table_entry
28 {
29 private:
30     data_set _entry_values;
31     data_type _expected_freq; //distribution-function dependent
32
33     /* This is not the min/max values of _entry_values
34      * but the interval's limits
35      */
36     data_type _min_value;
37     data_type _max_value;
38
39 public:
40     freq_table_entry(const data_set& entry_values, data_type min_value,
41             data_type max_value, data_type expected_freq = 0);
42
43     ~freq_table_entry() {};
44
45     data_type get_min_value() const;
46     data_type get_max_value() const;
47     unsigned int get_observed_freq() const;
48     data_type get_expected_freq() const;
49     data_set get_entry_values() const;
50
51     void set_expected_freq(data_type expected_freq);
52
53     void merge(const freq_table_entry& other_entry);
54 };
55 }
56
57 #endif /*FREQ_TABLE_ENTRY_H_*/
```

Listagem B.143: freq_table_entry.h

```
1 //
```

```
2 //     'FIT', a library for fitting statistical distribution
3 //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4 //
5 //     This library is free software; you can redistribute it and/or
6 //     modify it under the terms of the GNU Lesser General Public
7 //     License as published by the Free Software Foundation; either
8 //     version 2.1 of the License, or (at your option) any later version.
9 //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
```

```
19
20 #include <distribution/base_distrib.h>
21 #include <cstdlib>
22 #include <ctime>
23 #include <algorithm>
24
25 using namespace fit;
26
27 /*
28 base_distrib::base_distrib(const data_set& values)
29 {
30     _parameters = set_params_mle(values);
31     std::srand(std::time(0));
32 }
33 */
34
35
```

```
36 base_distrib::base_distrib()
37 {
38     std::srand(std::time(0));
39 }
40
41 base_distrib::base_distrib(const params_list& params)
42     : _parameters(params)
43 {
44     std::srand(std::time(0));
45 }
46
47 bool base_distrib::check_values(const data_set& values) const
48 {
49     for (data_set::const_iterator it = values.begin(); it != values.end();
          it++)
50         if (!in_range(*it))
51             return false;
52
53     return true;
54 }
55
56 data_type base_distrib::expected_freq(data_type from_number, data_type
       to_number, unsigned int total) const
57 {
58     return total * (cumulative_function(to_number) - cumulative_function(
          from_number));
59 }
60
61 data_type base_distrib::random_number() const
62 {
63     data_type ran = std::rand() / ( RAND_MAX + 1.0 );
64     if (ran == 0.0)
65         ran = 0.000000000001;
66     else if (ran == 1.0)
67         ran = 0.999999999999;
68     return ran;
69 }
```

Listagem B.144: base_distrib.cpp

```
1  //
   ------------------------------------------------------------------------

2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
   ------------------------------------------------------------------------

19
20 #ifndef BASE_DISTRIB_H_
21 #define BASE_DISTRIB_H_
22
23 #include "../fit_defs.h"
24 #include <vector>
25
26 namespace fit
27 {
28 typedef std::vector<data_type> params_list;
29 //the base probability distribution
30 class base_distrib
31 {
32 protected:
33     params_list _parameters;
34
```

```
35      /* Uniform(0;1) number generator */
36      data_type random_number() const;
37
38      /* Verifies weather value is in distribution possible range.
39       * In this case range is (-infinite, +infinite) */
40      virtual inline bool in_range(data_type value) const { return true; }
41
42      base_distrib();
43      base_distrib(const params_list& params); //gets distribution parameters
            from params.
44      //base_distrib(const data_set& values); //gets distribution parameters
            from MLE
45
46  public:
47
48      //Return a clone of de distribuition allocated on heap
49      virtual base_distrib *clone() = 0;
50
51      virtual std::string name() const = 0;
52      virtual std::vector<std::string> paramNames() const = 0;
53
54      virtual ~base_distrib() {};
55
56      virtual data_type get_mean() const = 0;
57      virtual data_type get_variance() const = 0;
58      virtual data_type get_mode() const
59      {
60          throw distribution_exception("Mode does not uniquely exist.");
61      }
62
63      /* expected_freq = total * (cumulative_function(to_number) -
            cumulative_function(from_number)) */
64      data_type expected_freq(data_type from_number, data_type to_number,
            unsigned int total = 1) const;
65
66      virtual data_type cumulative_function(data_type number) const = 0; //
            cumulative distribution function
67      virtual data_type density_function(data_type number) const = 0; //
            probability density function
68
69      /* Sets distribution parameters through maximum likelihood procedures
            */
70      virtual void set_params_mle(const data_set& values) = 0;
71
72      /* Gets a random value according to distribution */
73      virtual data_type get_random() const
74      {
75          throw distribution_exception("No random number generator defined
                for this function.");
76      }
77
78      virtual bool check_parameters(const params_list& params) const = 0;
79      bool check_values(const data_set& values) const;
80
81      unsigned int get_params_number() const { return _parameters.size(); }
82      params_list get_params() const { return _parameters; }
83
84  };
85  }
86
87  #endif /*BASE_DISTRIB_H_*/
```

Listagem B.145: base_distrib.h

```
1   //
2   //     'FIT', a library for fitting statistical distribution
3   //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4   //
5   //     This library is free software; you can redistribute it and/or
6   //     modify it under the terms of the GNU Lesser General Public
7   //     License as published by the Free Software Foundation; either
8   //     version 2.1 of the License, or (at your option) any later version.
9   //
10  //     This library is distributed in the hope that it will be useful,
11  //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12  //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13  //     Lesser General Public License for more details.
```

```
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
```

```
19
20 #include <distribution/beta_distrib.h>
21 #include <common/fit_utils.hpp>
22
23 using namespace fit;
24
25 beta_distrib::beta_distrib(const beta_distrib& other) :
26 gamm1(new gamm_distrib(*other.gamm1)),
27     gamm2(new gamm_distrib(*other.gamm2)),
28     base_distrib(other._parameters)
29 {
30 }
31
32
33 beta_distrib::beta_distrib(const data_set& values)
34 {
35     _parameters.resize(PARAMS_NUMBER);
36     set_params_mle(values);
37
38     params_list par;
39     par.push_back(_parameters[ALPHA1]);
40     par.push_back(1.0);
41     gamm1 = new gamm_distrib(par);
42
43     par[0] = _parameters[ALPHA2];
44     gamm2 = new gamm_distrib(par);
45 }
46
47 beta_distrib::beta_distrib(const params_list& params)
48 {
49     if (!check_parameters(params))
50         throw distribution_exception("Invalid parameters on Beta
              distribution.");
51
52     _parameters = params;
53
54     params_list par;
55     par.push_back(_parameters[ALPHA1]);
56     par.push_back(1.0);
57     gamm1 = new gamm_distrib(par);
58
59     par[0] = _parameters[ALPHA2];
60     gamm2 = new gamm_distrib(par);
61 }
62
63 base_distrib *beta_distrib::clone()
64 {
65     return new beta_distrib(*this);
66 }
67
68 std::string beta_distrib::name() const
69 {
70     return "Distribuição Beta";
71 }
72
73 std::vector<std::string> beta_distrib::paramNames() const
74 {
75     std::vector<std::string> paramNames = std::vector<std::string>();
76     paramNames.push_back("Alpha 1");
77     paramNames.push_back("Alpha 2");
78
79     return paramNames;
80 }
81
82 beta_distrib::~beta_distrib()
83 {
84     delete gamm1;
85     delete gamm2;
86 }
87
88 data_type beta_distrib::get_mean() const
89 {
90     data_type alpha = _parameters[ALPHA1];
91     return alpha / (alpha + _parameters[ALPHA2]);
```

```
92 }
93
94 data_type beta_distrib::get_variance() const
95 {
96     data_type alpha1 = _parameters[ALPHA1];
97     data_type alpha2 = _parameters[ALPHA2];
98     data_type den_a = (alpha1 + alpha2) * (alpha1 + alpha2);
99     return (alpha1 * alpha2) / ( den_a * (alpha1 + alpha2 + 1) );
100 }
101
102 data_type beta_distrib::get_mode() const
103 {
104     data_type alpha1 = _parameters[ALPHA1];
105     data_type alpha2 = _parameters[ALPHA2];
106
107     if ((alpha1 == alpha2) && (alpha1 == 1))
108         throw distribution_exception("Mode does not uniquely exist.");
109
110     if ((alpha1 < 1) && (alpha2 < 1))
111         return 0; // or return 1;
112
113     if (((alpha1 < 1) && (alpha2 >= 1)) || ((alpha1 == 1) && (alpha2 > 1)))
114         return 0;
115
116     if (((alpha1 >= 1) && (alpha2 < 1)) || ((alpha1 > 1) && (alpha2 == 1)))
117         return 1;
118
119     if ((alpha1 > 1) && (alpha2 > 1))
120         return (alpha1 - 1) / (alpha1 + alpha2 - 2);
121
122     return 0;
123 }
124
125 data_type beta_distrib::cumulative_function(data_type number) const
126 {
127     return utils::betai(_parameters[ALPHA1], _parameters[ALPHA2], number);
128 }
129
130 data_type beta_distrib::beta(data_type z, data_type w)
131 {
132     return std::exp(utils::gammln(z) + utils::gammln(w) - utils::gammln(z+w
          ));
133 }
134
135 data_type beta_distrib::density_function(data_type number) const
136 {
137     if ( (number <= 0) || (number >= 1) )
138         return 0;
139
140     data_type alpha1 = _parameters[ALPHA1];
141     data_type alpha2 = _parameters[ALPHA2];
142     data_type fracnum = std::pow(number, alpha1 - 1) * std::pow(1 - number,
           alpha2 - 1);
143     return fracnum / beta_distrib::beta(alpha1, alpha2);
144 }
145
146 void beta_distrib::set_params_mle(const data_set& values)
147 {
148     if (!check_values(values))
149         throw distribution_exception("Invalid value on Beta distribution.")
               ;
150
151     data_type media = utils::_mean(values);
152     data_type varian = utils::_variance(media, values);
153     _parameters[ALPHA1] = media * (((media * (1 - media))/varian) - 1);
154     _parameters[ALPHA2] = (1 - media) * (((media * (1 - media))/varian) -
          1);
155
156     if (!check_parameters(_parameters))
157         throw distribution_exception("Invalid parameters on Beta
               distribution.");
158 }
159
160 data_type beta_distrib::get_random() const
161 {
162     data_type Y1 = gamm1->get_random();
163     data_type Y2 = gamm2->get_random();
164     return Y1 / (Y1 + Y2);
165 }
166
167 bool beta_distrib::check_parameters(const params_list& params) const
168 {
```

```
169     return ((params.size() == PARAMS_NUMBER) &&
170             (params.at(ALPHA1) > 0) &&
171             (params.at(ALPHA2) > 0));
172 }
```

Listagem B.146: beta_distrib.cpp

```
1  //
2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
19
20 #ifndef BETA_DISTRIB_H_
21 #define BETA_DISTRIB_H_
22
23 /*
24  * Beta Distribution
25  */
26
27 #include "gamm_distrib.h"
28
29 namespace fit
30 {
31 class beta_distrib : public base_distrib
32 {
33 private:
34     gamm_distrib *gamm1, *gamm2;
35
36 protected:
37     inline bool in_range(data_type value) const { return (value >= 0) && (
          value <= 1); }
38     static data_type beta(data_type z, data_type w);
39
40 public:
41     enum _beta_params
42     {
43         ALPHA1,
44         ALPHA2,
45         PARAMS_NUMBER
46     };
47
48     beta_distrib(const beta_distrib& other);
49
50     beta_distrib(const data_set& values); //gets distribution parameters
          from MLE
51     beta_distrib(const params_list& params); //gets distribution parameters
          from params.
52
53     //Return a clone of de distribuition allocated on heap
54     base_distrib *clone();
55
56     std::string name() const;
57     std::vector<std::string> paramNames() const;
58
59     ~beta_distrib();
60
61     data_type get_mean() const;
62     data_type get_variance() const;
63     data_type get_mode() const;
64
```

```
65      data_type cumulative_function(data_type number) const; //cumulative
            distribution function
66      data_type density_function(data_type number) const; //probability
            density function
67
68      /* Sets distribution parameters through maximum likelihood procedures
            */
69      void set_params_mle(const data_set& values);
70
71      /* Gets a random value according to distribution */
72      data_type get_random() const;
73
74      bool check_parameters(const params_list& params) const;
75  };
76  }
77
78  #endif /*BETA_DISTRIB_H_*/
```

Listagem B.147: beta_distrib.h

```
1   //
2   //    'FIT', a library for fitting statistical distribution
3   //    Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4   //
5   //    This library is free software; you can redistribute it and/or
6   //    modify it under the terms of the GNU Lesser General Public
7   //    License as published by the Free Software Foundation; either
8   //    version 2.1 of the License, or (at your option) any later version.
9   //
10  //    This library is distributed in the hope that it will be useful,
11  //    but WITHOUT ANY WARRANTY; without even the implied warranty of
12  //    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13  //    Lesser General Public License for more details.
14  //
15  //    You should have received a copy of the GNU Lesser General Public
16  //    License along with this library; if not, write to the Free Software
17  //    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
        USA
18  //
19
20  #include <distribution/expo_distrib.h>
21  #include <cmath>
22  #include <common/fit_utils.hpp>
23
24  using namespace fit;
25  using std::exp;
26  using std::log;
27
28  expo_distrib::expo_distrib(const data_set& values)
29  {
30      _parameters.resize(PARAMS_NUMBER);
31      set_params_mle(values);
32  }
33
34  expo_distrib::expo_distrib(const params_list& params)
35  {
36      if (!check_parameters(params))
37          throw distribution_exception("Invalid parameters on exponential
                distribution.");
38
39      _parameters = params;
40  }
41
42  base_distrib *expo_distrib::clone()
43  {
44      return new expo_distrib(*this);
45  }
46
47  std::string expo_distrib::name() const
48  {
49      return "Distribuição Exponencial";
50  }
51
52  std::vector<std::string> expo_distrib::paramNames() const
53  {
```

```
54      std::vector<std::string> paramNames = std::vector<std::string>();
55      paramNames.push_back("Beta");
56
57      return paramNames;
58  }
59
60  data_type expo_distrib::get_mean() const
61  {
62      return _parameters[BETA];
63  }
64
65  data_type expo_distrib::get_variance() const
66  {
67      data_type beta = _parameters[BETA];
68      return beta * beta;
69  }
70
71  data_type expo_distrib::cumulative_function(data_type number) const
72  {
73      if (number < 0)
74          return 0;
75
76      return 1 - (exp( (number * -1) / _parameters[BETA]) );
77  }
78
79  data_type expo_distrib::density_function(data_type number) const
80  {
81      if (number < 0)
82          return 0;
83
84      data_type beta = _parameters[BETA];
85
86      return (1 / beta) * exp((-number) / beta);
87  }
88
89  void expo_distrib::set_params_mle(const data_set& values)
90  {
91      if (!check_values(values))
92          throw distribution_exception("Invalid value on exponencial
                distribution.");
93
94      _parameters[BETA] = utils::_mean(values);
95  }
96
97  data_type expo_distrib::get_random() const
98  {
99      return (-_parameters[BETA]) * log(1 - random_number());
100 }
101
102 bool expo_distrib::check_parameters(const params_list& params) const
103 {
104     return ((params.size() == PARAMS_NUMBER) && (params.at(BETA) > 0));
105 }
```

Listagem B.148: expo_distrib.cpp

```
1  //
   ------------------------------------------------------------------------

2  //      'FIT', a library for fitting statistical distribution
3  //      Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //      This library is free software; you can redistribute it and/or
6  //      modify it under the terms of the GNU Lesser General Public
7  //      License as published by the Free Software Foundation; either
8  //      version 2.1 of the License, or (at your option) any later version.
9  //
10 //      This library is distributed in the hope that it will be useful,
11 //      but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //      Lesser General Public License for more details.
14 //
15 //      You should have received a copy of the GNU Lesser General Public
16 //      License along with this library; if not, write to the Free Software
17 //      Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
        USA
18 //
   ------------------------------------------------------------------------
```

```
19
20 #ifndef EXPO_DISTRIB_H_
21 #define EXPO_DISTRIB_H_
22
23 /*
24  * Exponential Distribution
25  */
26
27 #include "base_distrib.h"
28
29 namespace fit
30 {
31 class expo_distrib : public base_distrib
32 {
33 protected:
34     inline bool in_range(data_type value) const { return value >= 0; }
35
36 public:
37     enum _expo_params
38     {
39         BETA,
40         PARAMS_NUMBER
41     };
42
43     expo_distrib(const data_set& values); //gets distribution parameters
          from MLE
44     expo_distrib(const params_list& params); //gets distribution parameters
           from params.
45
46     //Return a clone of de distribuition allocated on heap
47     base_distrib *clone();
48
49     std::string name() const;
50     std::vector<std::string> paramNames() const;
51
52     ~expo_distrib() {};
53
54     data_type get_mean() const;
55     data_type get_variance() const;
56     data_type get_mode() const { return 0; }
57
58     data_type cumulative_function(data_type number) const; //cumulative
          distribution function
59     data_type density_function(data_type number) const; //probability
          density function
60
61     /* Sets distribution parameters through maximum likelihood procedures
          */
62     void set_params_mle(const data_set& values);
63
64     /* Gets a random value according to distribution */
65     data_type get_random() const;
66
67     bool check_parameters(const params_list& params) const;
68 };
69 }
70
71 #endif /*EXPO_DISTRIB_H_*/
```

Listagem B.149: expo_distrib.h

```
1  //
2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
```

```
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //  ------------------------------------------------------------------------


19
20 #include <distribution/gamm_distrib.h>
21 #include <common/fit_utils.hpp>
22 #include <cmath>
23
24 using namespace fit;
25 using std::exp;
26 using std::pow;
27 using std::log;
28
29 gamm_distrib::gamm_distrib(const data_set& values)
30 {
31     _parameters.resize(PARAMS_NUMBER);
32     set_params_mle(values);
33 }
34
35 gamm_distrib::gamm_distrib(const params_list& params)
36 {
37     if (!check_parameters(params))
38         throw distribution_exception("Invalid parameters on Gamma
              distribution.");
39
40     _parameters = params;
41 }
42
43 base_distrib *gamm_distrib::clone()
44 {
45     return new gamm_distrib(*this);
46 }
47
48 std::string gamm_distrib::name() const
49 {
50     return "Distribuição Gamma";
51 }
52
53 std::vector<std::string> gamm_distrib::paramNames() const
54 {
55     std::vector<std::string> paramNames = std::vector<std::string>();
56     paramNames.push_back("Alpha");
57     paramNames.push_back("Beta");
58
59     return paramNames;
60 }
61
62 data_type gamm_distrib::get_mean() const
63 {
64     return _parameters[ALPHA] * _parameters[BETA];
65 }
66
67 data_type gamm_distrib::get_variance() const
68 {
69     data_type beta = _parameters[BETA];
70     return _parameters[ALPHA] * (beta * beta);
71 }
72
73 data_type gamm_distrib::get_mode() const
74 {
75     data_type alpha = _parameters[ALPHA];
76     if (alpha < 1)
77         return 0;
78
79     return _parameters[BETA] * (alpha - 1);
80 }
81
82 data_type gamm_distrib::cumulative_function(data_type number) const
83 {
84     return utils::gammp(_parameters[ALPHA], (number / _parameters[BETA]));
85 }
86
87 data_type gamm_distrib::density_function(data_type number) const
88 {
89     if (number <= 0)
90         return 0;
91
92     data_type alpha = _parameters[ALPHA];
93     data_type beta = _parameters[BETA];
94
```

```
95      data_type firstterm = pow(beta, -alpha) * pow(number, alpha - 1) * exp
            (-number / beta);
96      return firstterm / exp(utils::gammln(alpha));
97  }
98
99  void gamm_distrib::set_params_mle(const data_set& values)
100 {
101     if (!check_values(values))
102         throw distribution_exception("Invalid value on Gamma distribution."
                );
103
104     data_type mean = utils::_mean(values);
105     data_type variance = utils::_variance(mean, values);
106
107     _parameters[ALPHA] = (mean * mean) / variance;
108     _parameters[BETA] = variance / mean;
109
110     if (!check_parameters(_parameters))
111         throw distribution_exception("Invalid parameters on Gamma
                distribution.");
112 }
113
114 data_type gamm_distrib::get_random() const
115 {
116     data_type alpha = _parameters[ALPHA];
117     data_type beta = _parameters[BETA];
118
119     // according to Law and Kelton, Simulation Modeling and Analysis, 1991
120     // pages 488-489
121     if (alpha < 1.0)
122     {
123         data_type b = (exp(1.0) + alpha) / exp(1.0);
124         int counter = 0;
125         while (counter < 1000)
126         {
127             // step 1.
128             data_type P = b * random_number();
129             if (P <= 1.0){
130                 // step 2.
131                 data_type Y = pow(P, 1.0 / alpha);
132                 data_type U2 = random_number();
133                 if (U2 <= exp(-Y))
134                     return beta * Y;
135             }
136             else
137             {
138                 // step 3.
139                 data_type Y = -log((b - P) / alpha);
140                 data_type U2 = random_number();
141                 if (U2 <= pow(Y, alpha - 1.0))
142                     return beta * Y;
143             }
144             counter++;
145         }
146         return 1.0;
147     }
148     else if (alpha > 1.0)
149     {
150         // according to Law and Kelton, Simulation Modeling and Analysis,
                1991
151         // pages 488-489
152         data_type a = 1.0 / sqrt(2.0 * alpha - 1.0);
153         data_type b = alpha - log(4.0);
154         data_type q = alpha + (1.0 / a);
155         data_type theta = 4.5;
156         data_type d = 1.0 + log(theta);
157         int counter = 0;
158         while (counter < 1000){
159             // step 1.
160             data_type U1 = random_number();
161             data_type U2 = random_number();
162             // step 2.
163             data_type V = a * log(U1 / (1.0 - U1));
164             data_type Y = alpha * exp(V);
165             data_type Z = U1 * U1 * U2;
166             data_type W = b + q * V - Y;
167             // step 3.
168             if ((W + d - theta * Z) >= 0.0)
169                 return beta * Y;
170             else{
171                 // step 4.
172                 if (W > log(Z))
```

```
173                     return beta * Y;
174                 }
175                 counter++;
176             }
177             return 1.0;
178         }
179         else // alpha == 1.0
180         {
181             // Gamma(1.0, beta) ~ exponential with mean = beta
182             return -beta * log(random_number());
183         }
184 }
185
186 bool gamm_distrib::check_parameters(const params_list& params) const
187 {
188     return ((params.size() == PARAMS_NUMBER) &&
189             (params.at(ALPHA) > 0) &&
190             (params.at(BETA) > 0));
191 }
```

Listagem B.150: gamm_distrib.cpp

```
1  //
2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
19
20 #ifndef GAMM_DISTRIB_H_
21 #define GAMM_DISTRIB_H_
22
23 /*
24  * Gamma Distribution
25  */
26
27 #include "base_distrib.h"
28
29 namespace fit
30 {
31 class gamm_distrib : public base_distrib
32 {
33 protected:
34     inline bool in_range(data_type value) const { return value >= 0; }
35
36 public:
37     enum _gamm_params
38     {
39         ALPHA,
40         BETA,
41         PARAMS_NUMBER
42     };
43
44     gamm_distrib(const data_set& values); //gets distribution parameters
          from MLE
45     gamm_distrib(const params_list& params); //gets distribution parameters
          from params.
46
47     //Return a clone of de distribuition allocated on heap
48     base_distrib *clone();
49
50     std::string name() const;
```

```
51      std::vector<std::string> paramNames() const;
52
53      ~gamm_distrib() {};
54
55      data_type get_mean() const;
56      data_type get_variance() const;
57      data_type get_mode() const;
58
59      data_type cumulative_function(data_type number) const; //cumulative
            distribution function
60      data_type density_function(data_type number) const; //probability
            density function
61
62      /* Sets distribution parameters through maximum likelihood procedures
            */
63      void set_params_mle(const data_set& values);
64
65      /* Gets a random value according to distribution */
66      data_type get_random() const;
67
68      bool check_parameters(const params_list& params) const;
69  };
70  }
71
72  #endif /*GAMM_DISTRIB_H_*/
```

Listagem B.151: gamm_distrib.h

```
1   //
    ------------------------------------------------------------------------

2   //      'FIT', a library for fitting statistical distribution
3   //      Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4   //
5   //      This library is free software; you can redistribute it and/or
6   //      modify it under the terms of the GNU Lesser General Public
7   //      License as published by the Free Software Foundation; either
8   //      version 2.1 of the License, or (at your option) any later version.
9   //
10  //      This library is distributed in the hope that it will be useful,
11  //      but WITHOUT ANY WARRANTY; without even the implied warranty of
12  //      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13  //      Lesser General Public License for more details.
14  //
15  //      You should have received a copy of the GNU Lesser General Public
16  //      License along with this library; if not, write to the Free Software
17  //      Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
        USA
18  //
    ------------------------------------------------------------------------

19
20  #include <distribution/lognorm_distrib.h>
21  #include <cmath>
22  #include <algorithm>
23  #include <common/fit_utils.hpp>
24
25  using namespace fit;
26  using std::exp;
27  using std::log;
28  using std::sqrt;
29
30  data_set lognorm_distrib::get_log_values(const data_set& values)
31  {
32      data_set log_values;
33
34      //log_values.resize(values.size());
35      //std::transform(values.begin(), values.end(), log_values.begin(), std
            ::log);
36
37      for (data_set::const_iterator it = values.begin(); it != values.end();
            it++)
38          log_values.push_back( std::log(*it) );
39
40      return log_values;
41  }
42
43  base_distrib *lognorm_distrib::clone()
44  {
```

```
45      return new lognorm_distrib(*this);
46  }
47
48  std::string lognorm_distrib::name() const
49  {
50      return "Distribuição Log-normal";
51  }
52
53  std::vector<std::string> lognorm_distrib::paramNames() const
54  {
55      return std::vector<std::string>();
56  }
57
58  lognorm_distrib::lognorm_distrib(const data_set& values)
59  {
60      if (!check_values(values))
61          throw distribution_exception("Invalid value on lognormal
                distribution.");
62
63      _parameters.resize(PARAMS_NUMBER);
64      set_params_mle(get_log_values(values));
65  }
66
67  lognorm_distrib::lognorm_distrib(const params_list& params)
68      : norm_distrib(params)
69  {
70  }
71
72  data_type lognorm_distrib::get_mean() const
73  {
74      data_type sigma = _parameters[STDDEV];
75      return exp(_parameters[MEAN] + ((sigma * sigma) / 2));
76  }
77
78  data_type lognorm_distrib::get_variance() const
79  {
80      data_type sigma = _parameters[STDDEV];
81      data_type sigma_sqr = sigma * sigma;
82      return exp((2 * _parameters[MEAN]) + sigma_sqr) * (exp(sigma_sqr) - 1);
83  }
84
85  data_type lognorm_distrib::get_mode() const
86  {
87      data_type sigma = _parameters[STDDEV];
88      data_type sigma_sqr = sigma * sigma;
89      return exp(_parameters[MEAN] - sigma_sqr);
90  }
91
92  data_type lognorm_distrib::cumulative_function(data_type number) const
93  {
94      return norm_distrib::cumulative_function( std::log(number) );
95  }
96
97  data_type lognorm_distrib::density_function(data_type number) const
98  {
99      if (number <= 0)
100         return 0;
101
102     data_type sigma = _parameters[STDDEV];
103     data_type sigma_sqr = sigma * sigma;
104     data_type mu = _parameters[MEAN];
105
106     data_type numer = exp( (-pow(log(number) - mu, 2)) / (2 * sigma_sqr) );
107     data_type denom = number * sigma * sqrt(2 * utils::PI);
108
109     return numer / denom;
110 }
111
112 data_type lognorm_distrib::get_random() const
113 {
114     return exp(norm_distrib::get_random());
115 }
```

Listagem B.152: lognorm_distrib.cpp

```
1  //
2  //    'FIT', a library for fitting statistical distribution
```

```
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //  ------------------------------------------------------------------------

19
20 #ifndef LOGNORM_DISTRIB_H_
21 #define LOGNORM_DISTRIB_H_
22
23 /*
24  * Lognormal Distribution
25  */
26
27 #include "norm_distrib.h"
28
29 namespace fit
30 {
31 class lognorm_distrib : public norm_distrib
32 {
33 private:
34     static data_set get_log_values(const data_set& values);
35
36 protected:
37     inline bool in_range(data_type value) const { return value >= 0; }
38
39 public:
40     lognorm_distrib(const data_set& values); //gets distribution parameters
          from MLE
41     lognorm_distrib(const params_list& params); //gets distribution
          parameters from params.
42
43     //Return a clone of de distribuition allocated on heap
44     base_distrib *clone();
45
46     std::string name() const;
47     std::vector<std::string> paramNames() const;
48
49     ~lognorm_distrib() {};
50
51     data_type get_mean() const;
52     data_type get_variance() const;
53     data_type get_mode() const;
54
55     data_type cumulative_function(data_type number) const;
56     data_type density_function(data_type number) const; //probability
          density function
57
58     /* Sets distribution parameters through maximum likelihood procedures
          */
59     //void set_params_mle(const data_set& values);
60
61     /* Gets a random value according to distribution */
62     data_type get_random() const;
63 };
64 }
65
66 #endif /*LOGNORM_DISTRIB_H_*/
```

Listagem B.153: lognorm_distrib.h

```
1  //  ------------------------------------------------------------------------

2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
```

```
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
   ------------------------------------------------------------------------------

19
20 #include <distribution/norm_distrib.h>
21 #include <cmath>
22 #include <common/fit_utils.hpp>
23
24 using namespace fit;
25 using std::sqrt;
26 using std::log;
27 using std::sin;
28 using std::abs;
29 using std::exp;
30
31 norm_distrib::norm_distrib(const data_set& values)
32 {
33     _parameters.resize(PARAMS_NUMBER);
34     set_params_mle(values);
35 }
36
37 base_distrib *norm_distrib::clone()
38 {
39     return new norm_distrib(*this);
40 }
41
42 std::string norm_distrib::name() const
43 {
44     return "Distribuição Normal";
45 }
46
47 std::vector<std::string> norm_distrib::paramNames() const
48 {
49     std::vector<std::string> paramNames = std::vector<std::string>();
50     paramNames.push_back("Média");
51     paramNames.push_back("Desvio Padrão");
52
53     return paramNames;
54 }
55
56 norm_distrib::norm_distrib(const params_list& params)
57 {
58     if (!check_parameters(params))
59         throw distribution_exception("Invalid parameters on normal
              distribution.");
60
61     _parameters = params;
62 }
63
64 data_type norm_distrib::cumulative_function(data_type number) const
65 {
66     data_type z_number = ( number - _parameters[MEAN] ) / _parameters[
          STDDEV];
67     return normal_cdf(z_number);
68 }
69
70 data_type norm_distrib::density_function(data_type number) const
71 {
72     data_type mu = _parameters[MEAN];
73     data_type sigsqr = get_variance();
74     data_type numer = exp( ((number - mu) * (number - mu)) / (2 * sigsqr) )
          ;
75     data_type denom = sqrt(2 * sigsqr * utils::PI);
76
77     return numer / denom;
78 }
79
```

```
80  void norm_distrib::set_params_mle(const data_set& values)
81  {
82      data_type md = utils::_mean(values);
83      data_type sigma = sqrt(utils::_variance(md, values));
84
85      _parameters[MEAN] = md;
86      _parameters[STDDEV] = sigma;
87  }
88
89  data_type norm_distrib::get_mean() const
90  {
91      return _parameters[MEAN];
92  }
93
94  data_type norm_distrib::get_variance() const
95  {
96      return std::pow(_parameters[STDDEV], 2);
97  }
98
99  data_type norm_distrib::get_mode() const
100 {
101     return _parameters[MEAN];
102 }
103
104 data_type norm_distrib::get_random() const
105 {
106     data_type z = sqrt( -2 * log(random_number()) ) * sin( 6.2831853071 *
          random_number() );
107     return _parameters[MEAN] + (z * _parameters[STDDEV]);
108 }
109
110 bool norm_distrib::check_parameters(const params_list& params) const
111 {
112     return ((params.size() == PARAMS_NUMBER) && (params.at(STDDEV) > 0));
113 }
114
115 //-------------------------------------------
116 int norm_distrib::sgn(const data_type& x)
117 {
118     return x < 0.0 ? -1 : 1;
119 }
120
121 data_type norm_distrib::g(data_type x, data_type u)
122 {
123     return 0.5 + u * sgn(x);
124 }
125
126 data_type norm_distrib::normal_cdf(data_type x)
127 {
128     const data_type ONE_SQRT_2_PI = 0.39894228040143267793;
129
130     double y = abs(x);
131
132     if (y > 7.335)
133         return g(x);
134
135     if (y < 0.0001) {
136         long double u = y * ONE_SQRT_2_PI;
137         return g(x, u);
138     }
139
140     long double a = 0.0;
141     long double b = 1.0;
142     long double u = 1.0;
143     long double j = -1.0;
144     int n = int(4 + 4 * y);
145     long double z = 1.0 / (y * y);
146     long double v = 1.0 / (4 * n + 2);
147     long double d = 0;
148
149     for (int i = n; i > 0; --i) {
150         d = a - 8 * i * b * z;
151         u = u + d * j;
152         v = v + (d - b) / (4 * i - 2);
153         a = b;
154         b = d;
155         j = -j;
156     }
157
158     u = (-v * y * ONE_SQRT_2_PI) / (u * j + d / 2.0);
159     return g(x, u);
160 }
```

Listagem B.154: norm_distrib.cpp

```
1  //
   ------------------------------------------------------------------------

2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
   ------------------------------------------------------------------------

19
20 #ifndef NORM_DISTRIB_H_
21 #define NORM_DISTRIB_H_
22
23 /*
24  * Normal Distribution
25  */
26
27 #include "base_distrib.h"
28
29 namespace fit
30 {
31 class norm_distrib : public base_distrib
32 {
33 protected:
34     static inline int sgn(const data_type& x);
35     static inline data_type g(data_type x, data_type u = 0.5);
36     static data_type normal_cdf(data_type x); //normal's cumulative
          distribution function.
37
38     norm_distrib() {};
39
40 public:
41     enum _norm_params
42     {
43         MEAN,
44         STDDEV,
45         PARAMS_NUMBER
46     };
47     norm_distrib(const data_set& values); //gets distribution parameters
          from MLE
48     norm_distrib(const params_list& params); //gets distribution parameters
           from params.
49
50     //Return a clone of de distribuition allocated on heap
51     base_distrib *clone();
52
53     std::string name() const;
54     std::vector<std::string> paramNames() const;
55
56     virtual ~norm_distrib() {};
57
58     virtual data_type get_mean() const;
59     virtual data_type get_variance() const;
60     virtual data_type get_mode() const;
61
62     virtual data_type cumulative_function(data_type number) const;
63     virtual data_type density_function(data_type number) const; //
          probability density function
64
65     /* Sets distribution parameters through maximum likelihood procedures
          */
66     void set_params_mle(const data_set& values);
```

```
67
68      /* Gets a random value according to distribution */
69      virtual data_type get_random() const;
70
71      bool check_parameters(const params_list& params) const;
72  };
73  }
74
75  #endif /*NORM_DISTRIB_H_*/
```

Listagem B.155: norm_distrib.h

```
1   //
        ------------------------------------------------------------------------
2   //      'FIT', a library for fitting statistical distribution
3   //      Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4   //
5   //      This library is free software; you can redistribute it and/or
6   //      modify it under the terms of the GNU Lesser General Public
7   //      License as published by the Free Software Foundation; either
8   //      version 2.1 of the License, or (at your option) any later version.
9   //
10  //      This library is distributed in the hope that it will be useful,
11  //      but WITHOUT ANY WARRANTY; without even the implied warranty of
12  //      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13  //      Lesser General Public License for more details.
14  //
15  //      You should have received a copy of the GNU Lesser General Public
16  //      License along with this library; if not, write to the Free Software
17  //      Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
        USA
18  //
        ------------------------------------------------------------------------

19
20  #include <distribution/tria_distrib.h>
21  #include <algorithm>
22  #include <cmath>
23  #include <common/fit_utils.hpp>
24
25  using namespace fit;
26  using std::sqrt;
27
28  tria_distrib::tria_distrib(const data_set& values)
29  {
30      _parameters.resize(PARAMS_NUMBER);
31      set_params_mle(values);
32  }
33
34  tria_distrib::tria_distrib(const params_list& params)
35  {
36      if (!check_parameters(params))
37          throw distribution_exception("Invalid parameters on triangular
                distribution.");
38
39      _parameters = params;
40  }
41
42  base_distrib *tria_distrib::clone()
43  {
44      return new tria_distrib(*this);
45  }
46
47  std::string tria_distrib::name() const
48  {
49      return "Distribuição Triangular";
50  }
51
52  std::vector<std::string> tria_distrib::paramNames() const
53  {
54      std::vector<std::string> paramNames = std::vector<std::string>();
55      paramNames.push_back("Mínimo");
56      paramNames.push_back("Moda");
57      paramNames.push_back("Máximo");
58
59      return paramNames;
60  }
61
```

```
62 data_type tria_distrib::get_mean() const
63 {
64     return (_parameters[MIN] + _parameters[MODE] + _parameters[MAX]) / 3;
65 }
66
67 data_type tria_distrib::get_variance() const
68 {
69     data_type a = _parameters[MIN];
70     data_type c = _parameters[MODE];
71     data_type b = _parameters[MAX];
72
73     return ((a * a) + (b * b) + (c * c) - (a * b) - (a * c) - (b * c)) /
           18;
74 }
75
76 data_type tria_distrib::get_mode() const
77 {
78     return _parameters[MODE];
79 }
80
81 data_type tria_distrib::cumulative_function(data_type number) const
82 {
83     data_type result = 0.0;
84     data_type a = _parameters[MIN];
85     data_type c = _parameters[MODE];
86     data_type b = _parameters[MAX];
87
88     if (number < a)
89         result = 0;
90     else if ( (number >= a) && (number <= c) )
91         result = ( (number - a) * (number - a) ) / ( (b - a) * (c - a) );
92     else if ( (number > c) && (number <= b) )
93         result = 1 - ( ( (b - number) * (b - number) ) / ( ( b - a) * (b -
               c) ) );
94     else
95         result = 1;
96
97     return result;
98 }
99
100 data_type tria_distrib::density_function(data_type number) const
101 {
102     data_type a = _parameters[MIN];
103     data_type c = _parameters[MODE];
104     data_type b = _parameters[MAX];
105
106     if ( (number < a) || (number > b) )
107         return 0;
108
109     if (number <= c) //a <= number <= c
110         return 2 * (number - a) / ( (b - a) * (c - a) );
111     else //c < number <= b
112         return 2 * (b - number) / ( (b - a) * (b - c) );
113 }
114
115 void tria_distrib::set_params_mle(const data_set& values)
116 {
117     data_type mean = utils::_mean(values);
118     data_type min = *std::min_element(values.begin(), values.end());
119     data_type max = *std::max_element(values.begin(), values.end());
120
121     _parameters[MIN] = min;
122     _parameters[MAX] = max;
123
124     /* This method is only a poor estimator, nevertheless
125      * triangular distribution is a rough model.
126      * Czuber has a better method although its dependent
127      * of a frequency table */
128     _parameters[MODE] = (3 * mean) - (min + max);
129
130     if (!check_parameters(_parameters))
131         throw distribution_exception("Invalid parameters on triangular
               distribution.");
132 }
133
134 data_type tria_distrib::get_random() const
135 {
136     data_type result = 0.0;
137     data_type r = random_number();
138     data_type min = _parameters[MIN];
139     data_type max = _parameters[MAX];
140     data_type mode = _parameters[MODE];
```

```
141     data_type dx = (mode - min) / (max - min);
142
143     if(r > dx)
144         result = max - sqrt((1 - r) * (max - mode) * (max - min));
145     else
146         result = min + sqrt(r * (mode - min) * (max - min));
147
148     return result;
149 }
150
151 bool tria_distrib::check_parameters(const params_list& params) const
152 {
153     return ((params.size() == PARAMS_NUMBER) &&
154             (params.at(MIN) < params.at(MODE)) &&
155             (params.at(MODE) < params.at(MAX)));
156 }
```

Listagem B.156: tria_distrib.cpp

```
1  //
   ------------------------------------------------------------------------

2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
   ------------------------------------------------------------------------

19
20 #ifndef TRIA_DISTRIB_H_
21 #define TRIA_DISTRIB_H_
22
23 /*
24  * Triangular Distribution
25  */
26
27 #include "base_distrib.h"
28
29 namespace fit
30 {
31 class tria_distrib : public base_distrib
32 {
33 public:
34     enum _tria_params
35     {
36         MIN,
37         MODE,
38         MAX,
39         PARAMS_NUMBER
40     };
41
42     tria_distrib(const data_set& values); //gets distribution parameters
          from MLE
43     tria_distrib(const params_list& params); //gets distribution parameters
          from params.
44
45     //Return a clone of de distribuition allocated on heap
46     base_distrib *clone();
47
48     std::string name() const;
49     std::vector<std::string> paramNames() const;
50
51     ~tria_distrib() {};
52
53     data_type get_mean() const;
```

```
54      data_type get_variance() const;
55      data_type get_mode() const;
56
57      data_type cumulative_function(data_type number) const; //cumulative
            distribution function
58      data_type density_function(data_type number) const; //probability
            density function
59
60      /* Sets distribution parameters through maximum likelihood procedures
            */
61      void set_params_mle(const data_set& values);
62
63      /* Gets a random value according to distribution */
64      data_type get_random() const;
65
66      bool check_parameters(const params_list& params) const;
67  };
68  }
69
70  #endif /*TRIA_DISTRIB_H_*/
```

Listagem B.157: tria_distrib.h

```
1   //
    ------------------------------------------------------------------------

2   //      'FIT', a library for fitting statistical distribution
3   //      Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4   //
5   //      This library is free software; you can redistribute it and/or
6   //      modify it under the terms of the GNU Lesser General Public
7   //      License as published by the Free Software Foundation; either
8   //      version 2.1 of the License, or (at your option) any later version.
9   //
10  //      This library is distributed in the hope that it will be useful,
11  //      but WITHOUT ANY WARRANTY; without even the implied warranty of
12  //      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13  //      Lesser General Public License for more details.
14  //
15  //      You should have received a copy of the GNU Lesser General Public
16  //      License along with this library; if not, write to the Free Software
17  //      Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
        USA
18  //
    ------------------------------------------------------------------------

19
20  #include <distribution/unif_distrib.h>
21  #include <algorithm>
22
23  using namespace fit;
24
25  unif_distrib::unif_distrib(const data_set& values)
26  {
27      _parameters.resize(PARAMS_NUMBER);
28      set_params_mle(values);
29  }
30
31  unif_distrib::unif_distrib(const params_list& params)
32  {
33      if (!check_parameters(params))
34          throw distribution_exception("Invalid parameters on uniform
                distribution.");
35
36      _parameters = params;
37  }
38
39  base_distrib *unif_distrib::clone()
40  {
41      return new unif_distrib(*this);
42  }
43
44  std::string unif_distrib::name() const
45  {
46      return "Distribuição Uniforme";
47  }
48
49  std::vector<std::string> unif_distrib::paramNames() const
50  {
```

```
51      std::vector<std::string> paramNames = std::vector<std::string>();
52      paramNames.push_back("Mínimo");
53      paramNames.push_back("Máximo");
54
55      return paramNames;
56  }
57
58  data_type unif_distrib::get_mean() const
59  {
60      return (_parameters[MIN] + _parameters[MAX]) / 2.0;
61  }
62
63  data_type unif_distrib::get_variance() const
64  {
65      data_type a = _parameters[MIN];
66      data_type b = _parameters[MAX];
67
68      return ((b - a) * (b - a)) / 12;
69  }
70
71  data_type unif_distrib::cumulative_function(data_type number) const
72  {
73      data_type result = 0.0;
74      data_type a = _parameters[MIN];
75      data_type b = _parameters[MAX];
76
77      if (number < a)
78          result = 0.0;
79      else if (number > b)
80          result = 1.0;
81      else
82          result = (number - a) / (b - a);
83
84      return result;
85  }
86
87  data_type unif_distrib::density_function(data_type number) const
88  {
89      data_type a = _parameters[MIN];
90      data_type b = _parameters[MAX];
91
92      if ( (number < a) || (number > b) )
93          return 0;
94
95      return 1 / (b - a);
96  }
97
98  void unif_distrib::set_params_mle(const data_set& values)
99  {
100     _parameters[MIN] = *std::min_element(values.begin(),values.end());
101     _parameters[MAX] = *std::max_element(values.begin(),values.end());
102
103     if (!check_parameters(_parameters))
104         throw distribution_exception("Invalid parameters on uniform
                distribution.");
105 }
106
107 data_type unif_distrib::get_random() const
108 {
109     data_type a = _parameters[MIN];
110     data_type b = _parameters[MAX];
111     return a + ((b - a) * random_number());
112 }
113
114 bool unif_distrib::check_parameters(const params_list& params) const
115 {
116     return ((params.size() == PARAMS_NUMBER) && (params.at(MAX) > params.at
            (MIN)));
117 }
```

Listagem B.158: unif_distrib.cpp

```
1  //
2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
```

```
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
   ------------------------------------------------------------------------

19
20 #ifndef UNIF_DISTRIB_H_
21 #define UNIF_DISTRIB_H_
22
23 /*
24  * Uniform Distribution
25  */
26
27 #include "base_distrib.h"
28
29 namespace fit
30 {
31 class unif_distrib : public base_distrib
32 {
33 public:
34     enum _unif_params
35     {
36         MIN,
37         MAX,
38         PARAMS_NUMBER
39     };
40
41     unif_distrib(const data_set& values); //gets distribution parameters
          from MLE
42     unif_distrib(const params_list& params); //gets distribution parameters
          from params.
43
44     //Return a clone of de distribuition allocated on heap
45     base_distrib *clone();
46
47     std::string name() const;
48     std::vector<std::string> paramNames() const;
49
50     ~unif_distrib() {};
51
52     data_type get_mean() const;
53     data_type get_variance() const;
54
55     data_type cumulative_function(data_type number) const; //cumulative
          distribution function
56     data_type density_function(data_type number) const; //probability
          density function
57
58     /* Sets distribution parameters through maximum likelihood procedures
          */
59     void set_params_mle(const data_set& values);
60
61     /* Gets a random value according to distribution */
62     data_type get_random() const;
63
64     bool check_parameters(const params_list& params) const;
65 };
66 }
67
68 #endif /*UNIF_DISTRIB_H_*/
```

Listagem B.159: unif_distrib.h

```
1  //
   ------------------------------------------------------------------------

2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
```

```
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //   ------------------------------------------------------------------------

19
20 #include <distribution/weib_distrib.h>
21 #include <common/fit_utils.hpp>
22 #include <cmath>
23
24 using namespace fit;
25 using std::pow;
26 using std::exp;
27 using std::log;
28
29 weib_distrib::weib_distrib(const data_set& values)
30 {
31     _parameters.resize(PARAMS_NUMBER);
32     set_params_mle(values);
33 }
34
35 weib_distrib::weib_distrib(const params_list& params)
36 {
37     if (!check_parameters(params))
38         throw distribution_exception("Invalid parameters on weibull
              distribution.");
39
40     _parameters = params;
41 }
42
43 base_distrib *weib_distrib::clone()
44 {
45     return new weib_distrib(*this);
46 }
47
48 std::string weib_distrib::name() const
49 {
50     return "Distribuição de Weibull";
51 }
52
53 std::vector<std::string> weib_distrib::paramNames() const
54 {
55     std::vector<std::string> paramNames = std::vector<std::string>();
56     paramNames.push_back("Alpha");
57     paramNames.push_back("Beta");
58
59     return paramNames;
60 }
61
62
63 data_type weib_distrib::get_mean() const
64 {
65     data_type alpha = _parameters[ALPHA];
66     data_type gamm_ret = exp(utils::gammln(1 / alpha));
67     return gamm_ret * (_parameters[BETA] / alpha);
68 }
69
70 data_type weib_distrib::get_variance() const
71 {
72     data_type alpha = _parameters[ALPHA];
73     data_type beta = _parameters[BETA];
74     data_type gamm_a = exp(utils::gammln(1 / alpha));
75     data_type gamm_b = exp(utils::gammln(2 / alpha));
76     data_type sec_calc = (1 / alpha) * (gamm_a * gamm_a);
77
78     return ((beta * beta) / alpha) * ((2 * gamm_b) - sec_calc);
79 }
80
81 data_type weib_distrib::get_mode() const
```

```
82 {
83     data_type alpha = _parameters[ALPHA];
84 
85     if (alpha < 1)
86         return 0;
87 
88     return _parameters[BETA] * pow((alpha - 1) / alpha, (1 / alpha));
89 }
90 
91 data_type weib_distrib::cumulative_function(data_type number) const
92 {
93     if (number <= 0)
94         return 0;
95 
96     return 1 - exp((-1) * pow( (number / _parameters[BETA]), _parameters[
           ALPHA]) );
97 }
98 
99 data_type weib_distrib::density_function(data_type number) const
100 {
101     if (number <=0)
102         return 0;
103 
104     data_type alpha = _parameters[ALPHA];
105     data_type beta = _parameters[BETA];
106     return pow(alpha * beta, -alpha) * pow(number, alpha - 1) * exp(-pow(
           number/beta, alpha));
107 }
108 
109 void weib_distrib::set_params_mle(const data_set& values)
110 {
111     if (!check_values(values))
112         throw distribution_exception("Invalid value on weibull distribution
                .");
113 
114     using utils::PI;
115     using utils::EPS;
116     using utils::ITMAX;
117 
118     data_type values_num = values.size();
119 
120     data_set::const_iterator i;
121 
122     data_type temp = 0.0;
123     data_type sumValuesln = 0.0;
124     data_type sumValueslnSqr = 0.0;
125 
126     for (i = values.begin(); i != values.end(); i++)
127     {
128         temp = log(*i);
129         sumValuesln += temp;
130         sumValueslnSqr += (temp * temp);
131     }
132 
133     data_type constA = sumValuesln / values_num;
134     data_type valueB = 0.0;
135     data_type valueC = 0.0;
136     data_type valueH = 0.0;
137 
138     data_type old_alpha = 0.0;
139     data_type new_alpha = 0.0;
140 
141     /*Estimativa do primeiro old_alpha*/
142     old_alpha = sqrt((values_num-1)/((6/(PI*PI))*(sumValueslnSqr-((
           sumValuesln*sumValuesln)/values_num))));
143 
144     int j = 0;
145     for (j = 1; j <= ITMAX; j++)
146     {
147         for (i = values.begin(); i != values.end(); i++)
148         {
149             data_type valueLOG = log(*i);
150             data_type valuePOW = pow(*i,old_alpha);
151             valueB += valuePOW;
152             valueC += valuePOW * valueLOG;
153             valueH += valuePOW * (valueLOG * valueLOG);
154         }
155         new_alpha = old_alpha + ((constA + (1/old_alpha) - (valueC/valueB))
                / ((1/(old_alpha*old_alpha)) + (((valueB*valueH) - (valueC*
               valueC))/(valueB*valueB))));
156 
157         if ((new_alpha - old_alpha) < EPS)
```

```
158             {
159                 break;
160             }
161             else
162             {
163                 valueB = 0.0;
164                 valueC = 0.0;
165                 valueH = 0.0;
166                 old_alpha = new_alpha;
167             }
168         }
169         if (j > ITMAX)
170             throw distribution_exception("It was not possible to estimate
                    weibull's parameters");
171
172         _parameters[ALPHA] = new_alpha;
173         _parameters[BETA] = pow((valueB/values_num),(1/new_alpha));
174
175         if (!check_parameters(_parameters))
176             throw distribution_exception("Invalid parameters on weibull
                    distribution.");
177 }
178
179 data_type weib_distrib::get_random() const
180 {
181     data_type alpha = _parameters[ALPHA];
182     data_type beta = _parameters[BETA];
183
184     return beta * pow(-log(random_number()), (1 / alpha)) ;
185 }
186
187 bool weib_distrib::check_parameters(const params_list& params) const
188 {
189     return ((params.size() == PARAMS_NUMBER) &&
190             (params.at(ALPHA) > 0) &&
191             (params.at(BETA) > 0));
192 }
```

Listagem B.160: weib_distrib.cpp

```
1  //
   ------------------------------------------------------------------------

2  //     'FIT', a library for fitting statistical distribution
3  //     Copyright (C) 2007 Sanjay Formighieri < sanjayfm at gmail dot com >
4  //
5  //     This library is free software; you can redistribute it and/or
6  //     modify it under the terms of the GNU Lesser General Public
7  //     License as published by the Free Software Foundation; either
8  //     version 2.1 of the License, or (at your option) any later version.
9  //
10 //     This library is distributed in the hope that it will be useful,
11 //     but WITHOUT ANY WARRANTY; without even the implied warranty of
12 //     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
13 //     Lesser General Public License for more details.
14 //
15 //     You should have received a copy of the GNU Lesser General Public
16 //     License along with this library; if not, write to the Free Software
17 //     Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
       USA
18 //
   ------------------------------------------------------------------------

19
20 #ifndef WEIB_DISTRIB_H_
21 #define WEIB_DISTRIB_H_
22
23 /*
24  * Weibull Distribution
25  */
26
27 #include "base_distrib.h"
28
29 namespace fit
30 {
31 class weib_distrib : public base_distrib
32 {
33 protected:
34     inline bool in_range(data_type value) const {return value >= 0; }
```

```
35
36 public:
37     enum _weib_params
38     {
39         ALPHA,
40         BETA,
41         PARAMS_NUMBER
42     };
43
44     weib_distrib(const data_set& values); //gets distribution parameters
           from MLE
45     weib_distrib(const params_list& params); //gets distribution parameters
           from params.
46
47     //Return a clone of de distribuition allocated on heap
48     base_distrib *clone();
49
50     std::string name() const;
51     std::vector<std::string> paramNames() const;
52
53     ~weib_distrib() {};
54
55     data_type get_mean() const;
56     data_type get_variance() const;
57     data_type get_mode() const;
58
59     data_type cumulative_function(data_type number) const; //cumulative
           distribution function
60     data_type density_function(data_type number) const; //probability
           density function
61
62     /* Sets distribution parameters through maximum likelihood procedures
           */
63     void set_params_mle(const data_set& values);
64
65     /* Gets a random value according to distribution */
66     data_type get_random() const;
67
68     bool check_parameters(const params_list& params) const;
69
70 };
71 }
72
73 #endif /*WEIB_DISTRIB_H_*/
```

Listagem B.161: weib_distrib.h

B.4 SUBPROJETO LIB_UTILIDADES

```
1  ï»¿<?xml version="1.0" encoding="utf-8"?>
2  <Project DefaultTargets="Build" ToolsVersion="4.0" xmlns="http://schemas.
       microsoft.com/developer/msbuild/2003">
3    <ItemGroup Label="ProjectConfigurations">
4      <ProjectConfiguration Include="Debug|Win32">
5        <Configuration>Debug</Configuration>
6        <Platform>Win32</Platform>
7      </ProjectConfiguration>
8      <ProjectConfiguration Include="Release|Win32">
9        <Configuration>Release</Configuration>
10       <Platform>Win32</Platform>
11     </ProjectConfiguration>
12   </ItemGroup>
13   <ItemGroup>
14     <ClCompile Include="GeneratedFiles\Debug\moc_dialog_progresso.cpp">
15       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
             Release|Win32'">true</ExcludedFromBuild>
16     </ClCompile>
17     <ClCompile Include="GeneratedFiles\Debug\moc_thread_controller.cpp">
18       <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
             Release|Win32'">true</ExcludedFromBuild>
19     </ClCompile>
20     <ClCompile Include="GeneratedFiles\Debug\moc_thread_controller_dialog.
           cpp">
```

```
21        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
22      </ClCompile>
23      <ClCompile Include="GeneratedFiles\Debug\moc_thread_worker.cpp">
24        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
25      </ClCompile>
26      <ClCompile Include="GeneratedFiles\Debug\moc_widget_progresso.cpp">
27        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='
              Release|Win32'">true</ExcludedFromBuild>
28      </ClCompile>
29      <ClCompile Include="GeneratedFiles\Release\moc_dialog_progresso.cpp">
30        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
31      </ClCompile>
32      <ClCompile Include="GeneratedFiles\Release\moc_thread_controller.cpp">
33        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
34      </ClCompile>
35      <ClCompile Include="GeneratedFiles\Release\moc_thread_controller_dialog
            .cpp">
36        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
37      </ClCompile>
38      <ClCompile Include="GeneratedFiles\Release\moc_thread_worker.cpp">
39        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
40      </ClCompile>
41      <ClCompile Include="GeneratedFiles\Release\moc_widget_progresso.cpp">
42        <ExcludedFromBuild Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">true</ExcludedFromBuild>
43      </ClCompile>
44      <ClCompile Include="thread_controller.cpp" />
45      <ClCompile Include="thread_controller_dialog.cpp" />
46      <ClCompile Include="utilidades.cpp" />
47    </ItemGroup>
48    <ItemGroup>
49      <ClInclude Include="IObserver.h" />
50      <ClInclude Include="utilidades.h" />
51      <CustomBuild Include="thread_controller_dialog.h">
52        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
53        <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Moc%27ing thread_controller_dialog.h...</Message>
54        <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
              >
55        <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DQT_CORE_LIB -DLIB_UTILIDADES_LIB -DQT_WIDGETS_LIB "-I
              .\GeneratedFiles" "-I." "-I$(QTDIR)\include" "-I.\GeneratedFiles
              \$(ConfigurationName)\." "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)
              \include\QtWidgets"</Command>
56        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
57        <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Moc%27ing thread_controller_dialog.h...</Message>
58        <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
              Outputs>
59        <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
              DLIB_UTILIDADES_LIB -DQT_WIDGETS_LIB "-I.\GeneratedFiles" "-I."
              "-I$(QTDIR)\include" "-I.\GeneratedFiles\$(ConfigurationName)\."
               "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtWidgets"</
              Command>
60      </CustomBuild>
61      <CustomBuild Include="widget_progresso.h">
62        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
              Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
63        <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
              Moc%27ing widget_progresso.h...</Message>
64        <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
              >
65        <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DQT_CORE_LIB -DLIB_UTILIDADES_LIB "-I.\GeneratedFiles"
```

```
            "-I." "-I$(QTDIR)\include" "-I.\GeneratedFiles\$(
            ConfigurationName)\." "-I$(QTDIR)\include\QtCore"</Command>
66        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
67        <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing widget_progresso.h...</Message>
68        <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
69        <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DLIB_UTILIDADES_LIB "-I.\GeneratedFiles" "-I." "-I$(QTDIR)\
            include" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I$(QTDIR)
            \include\QtCore"</Command>
70      </CustomBuild>
71      <CustomBuild Include="thread_worker.h">
72        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
73        <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing thread_worker.h...</Message>
74        <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
75        <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DQT_CORE_LIB -DLIB_UTILIDADES_LIB "-I.\GeneratedFiles"
            "-I." "-I$(QTDIR)\include" "-I.\GeneratedFiles\$(
            ConfigurationName)\." "-I$(QTDIR)\include\QtCore"</Command>
76        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
77        <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing thread_worker.h...</Message>
78        <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
79        <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DLIB_UTILIDADES_LIB "-I.\GeneratedFiles" "-I." "-I$(QTDIR)\
            include" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I$(QTDIR)
            \include\QtCore"</Command>
80      </CustomBuild>
81      <CustomBuild Include="thread_controller.h">
82        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
83        <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing thread_controller.h...</Message>
84        <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
            GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
            >
85        <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
            (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DQT_CORE_LIB -DLIB_UTILIDADES_LIB -DQT_WIDGETS_LIB "-I
            .\GeneratedFiles" "-I." "-I$(QTDIR)\include" "-I.\GeneratedFiles
            \$(ConfigurationName)\." "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)
            \include\QtWidgets"</Command>
86        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
            |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
87        <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            Moc%27ing thread_controller.h...</Message>
88        <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
            >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
            Outputs>
89        <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
            "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
            ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
            DQT_DLL -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
            DLIB_UTILIDADES_LIB -DQT_WIDGETS_LIB "-I.\GeneratedFiles" "-I."
             "-I$(QTDIR)\include" "-I.\GeneratedFiles\$(ConfigurationName)\."
             "-I$(QTDIR)\include\QtCore" "-I$(QTDIR)\include\QtWidgets"</
            Command>
90      </CustomBuild>
91      <CustomBuild Include="dialog_progresso.h">
92        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
93        <Message Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">
            Moc%27ing dialog_progresso.h...</Message>
```

```
94        <Outputs Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">.\
              GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</Outputs
              >
95        <Command Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">"$
              (QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DQT_CORE_LIB -DLIB_UTILIDADES_LIB "-I.\GeneratedFiles"
              "-I." "-I$(QTDIR)\include" "-I.\GeneratedFiles\$(
              ConfigurationName)\." "-I$(QTDIR)\include\QtCore"</Command>
96        <AdditionalInputs Condition="'$(Configuration)|$(Platform)'=='Release
              |Win32'">$(QTDIR)\bin\moc.exe;%(FullPath)</AdditionalInputs>
97        <Message Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              Moc%27ing dialog_progresso.h...</Message>
98        <Outputs Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
              >.\GeneratedFiles\$(ConfigurationName)\moc_%(Filename).cpp</
              Outputs>
99        <Command Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">
              "$(QTDIR)\bin\moc.exe" "%(FullPath)" -o ".\GeneratedFiles\$(
              ConfigurationName)\moc_%(Filename).cpp" -DUNICODE -DWIN32 -
              DQT_DLL -DQT_NO_DEBUG -DNDEBUG -DQT_CORE_LIB -
              DLIB_UTILIDADES_LIB "-I.\GeneratedFiles" "-I." "-I$(QTDIR)\
              include" "-I.\GeneratedFiles\$(ConfigurationName)\." "-I$(QTDIR)
              \include\QtCore"</Command>
100     </CustomBuild>
101   </ItemGroup>
102   <PropertyGroup Label="Globals">
103     <ProjectGuid>{2ABAEF46-6724-4740-9001-EE3D1760F5D7}</ProjectGuid>
104     <Keyword>Qt4VSv1.0</Keyword>
105   </PropertyGroup>
106   <Import Project="$(VCTargetsPath)\Microsoft.Cpp.Default.props" />
107   <PropertyGroup Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'"
          Label="Configuration">
108     <ConfigurationType>StaticLibrary</ConfigurationType>
109   </PropertyGroup>
110   <PropertyGroup Condition="'$(Configuration)|$(Platform)'=='Release|Win32'
          " Label="Configuration">
111     <ConfigurationType>StaticLibrary</ConfigurationType>
112   </PropertyGroup>
113   <Import Project="$(VCTargetsPath)\Microsoft.Cpp.props" />
114   <ImportGroup Label="ExtensionSettings">
115   </ImportGroup>
116   <ImportGroup Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'"
          Label="PropertySheets">
117     <Import Project="$(UserRootDir)\Microsoft.Cpp.$(Platform).user.props"
            Condition="exists('$(UserRootDir)\Microsoft.Cpp.$(Platform).user.
            props')" Label="LocalAppDataPlatform" />
118   </ImportGroup>
119   <ImportGroup Condition="'$(Configuration)|$(Platform)'=='Release|Win32'"
          Label="PropertySheets">
120     <Import Project="$(UserRootDir)\Microsoft.Cpp.$(Platform).user.props"
            Condition="exists('$(UserRootDir)\Microsoft.Cpp.$(Platform).user.
            props')" Label="LocalAppDataPlatform" />
121   </ImportGroup>
122   <PropertyGroup Label="UserMacros" />
123   <PropertyGroup>
124     <_ProjectFileVersion>10.0.30319.1</_ProjectFileVersion>
125     <CodeAnalysisRuleSet Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'">AllRules.ruleset</CodeAnalysisRuleSet>
126     <CodeAnalysisRules Condition="'$(Configuration)|$(Platform)'=='Debug|
            Win32'" />
127     <CodeAnalysisRuleAssemblies Condition="'$(Configuration)|$(Platform)
            '=='Debug|Win32'" />
128     <CodeAnalysisRuleSet Condition="'$(Configuration)|$(Platform)'=='
            Release|Win32'">AllRules.ruleset</CodeAnalysisRuleSet>
129     <CodeAnalysisRules Condition="'$(Configuration)|$(Platform)'=='Release|
            Win32'" />
130     <CodeAnalysisRuleAssemblies Condition="'$(Configuration)|$(Platform)
            '=='Release|Win32'" />
131     <OutDir Condition="'$(Configuration)|$(Platform)'=='Debug|Win32'">$(
            SolutionDir)$(Platform)\$(Configuration)\</OutDir>
132     <OutDir Condition="'$(Configuration)|$(Platform)'=='Release|Win32'">$(
            SolutionDir)$(Platform)\$(Configuration)\</OutDir>
133   </PropertyGroup>
134   <ItemDefinitionGroup Condition="'$(Configuration)|$(Platform)'=='Debug|
          Win32'">
135     <ClCompile>
136       <PreprocessorDefinitions>UNICODE;WIN32;QT_DLL;QT_CORE_LIB;
              LIB_UTILIDADES_LIB;QT_WIDGETS_LIB;%(PreprocessorDefinitions)</
              PreprocessorDefinitions>
137       <AdditionalIncludeDirectories>.\GeneratedFiles;.;$(QTDIR)\include;.\
              GeneratedFiles\$(ConfigurationName);$(QTDIR)\include\QtCore;$(
              QTDIR)\include\QtWidgets;%(AdditionalIncludeDirectories)</
```

```
            AdditionalIncludeDirectories>
138       <Optimization>Disabled</Optimization>
139       <DebugInformationFormat>ProgramDatabase</DebugInformationFormat>
140       <RuntimeLibrary>MultiThreadedDebugDLL</RuntimeLibrary>
141       <TreatWChar_tAsBuiltInType>false</TreatWChar_tAsBuiltInType>
142     </ClCompile>
143     <Link>
144       <SubSystem>Windows</SubSystem>
145       <OutputFile>$(OutDir)\$(ProjectName).dll</OutputFile>
146       <AdditionalLibraryDirectories>$(QTDIR)\lib;%(
            AdditionalLibraryDirectories)</AdditionalLibraryDirectories>
147       <GenerateDebugInformation>true</GenerateDebugInformation>
148       <AdditionalDependencies>qtmaind.lib;Qt5Cored.lib;%(
            AdditionalDependencies)</AdditionalDependencies>
149     </Link>
150   </ItemDefinitionGroup>
151   <ItemDefinitionGroup Condition="'$(Configuration)|$(Platform)'=='Release|
        Win32'">
152     <ClCompile>
153       <PreprocessorDefinitions>UNICODE;WIN32;QT_DLL;QT_NO_DEBUG;NDEBUG;
            QT_CORE_LIB;LIB_UTILIDADES_LIB;QT_WIDGETS_LIB;%(
            PreprocessorDefinitions)</PreprocessorDefinitions>
154       <AdditionalIncludeDirectories>.\GeneratedFiles;.;$(QTDIR)\include;.\
            GeneratedFiles\$(ConfigurationName);$(QTDIR)\include\QtCore;$(
            QTDIR)\include\QtWidgets;%(AdditionalIncludeDirectories)</
            AdditionalIncludeDirectories>
155       <DebugInformationFormat>
156       </DebugInformationFormat>
157       <RuntimeLibrary>MultiThreadedDLL</RuntimeLibrary>
158       <TreatWChar_tAsBuiltInType>false</TreatWChar_tAsBuiltInType>
159     </ClCompile>
160     <Link>
161       <SubSystem>Windows</SubSystem>
162       <OutputFile>$(OutDir)\$(ProjectName).dll</OutputFile>
163       <AdditionalLibraryDirectories>$(QTDIR)\lib;%(
            AdditionalLibraryDirectories)</AdditionalLibraryDirectories>
164       <GenerateDebugInformation>false</GenerateDebugInformation>
165       <AdditionalDependencies>qtmain.lib;Qt5Core.lib;%(
            AdditionalDependencies)</AdditionalDependencies>
166     </Link>
167   </ItemDefinitionGroup>
168   <Import Project="$(VCTargetsPath)\Microsoft.Cpp.targets" />
169   <ImportGroup Label="ExtensionTargets">
170   </ImportGroup>
171   <ProjectExtensions>
172     <VisualStudio>
173       <UserProperties UicDir=".\GeneratedFiles" MocDir=".\GeneratedFiles\$(
            ConfigurationName)" MocOptions="" RccDir=".\GeneratedFiles"
            lupdateOnBuild="0" lupdateOptions="" lreleaseOptions=""
            Qt5Version_x0020_Win32="msvc2010_opengl" />
174     </VisualStudio>
175   </ProjectExtensions>
176 </Project>
```

Listagem B.162: lib_utilidades.vcxproj

```
1 ï»¿<?xml version="1.0" encoding="utf-8"?>
2 <Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/
      msbuild/2003">
3   <ItemGroup>
4     <Filter Include="Source Files">
5       <UniqueIdentifier>{4FC737F1-C7A5-4376-A066-2A32D752A2FF}</
            UniqueIdentifier>
6       <Extensions>cpp;cxx;c;def</Extensions>
7     </Filter>
8     <Filter Include="Header Files">
9       <UniqueIdentifier>{93995380-89BD-4b04-88EB-625FBE52EBFB}</
            UniqueIdentifier>
10      <Extensions>h</Extensions>
11    </Filter>
12    <Filter Include="Form Files">
13      <UniqueIdentifier>{99349809-55BA-4b9d-BF79-8FDBB0286EB3}</
            UniqueIdentifier>
14      <Extensions>ui</Extensions>
15    </Filter>
16    <Filter Include="Resource Files">
17      <UniqueIdentifier>{D9D6E242-F8AF-46E4-B9FD-80ECBC20BA3E}</
            UniqueIdentifier>
18      <Extensions>qrc;*</Extensions>
```

```
19        <ParseFiles>false</ParseFiles>
20      </Filter>
21      <Filter Include="Generated Files">
22        <UniqueIdentifier>{71ED8ED8-ACB9-4CE9-BBE1-E00B30144E11}</
              UniqueIdentifier>
23        <Extensions>moc;h;cpp</Extensions>
24        <SourceControlFiles>False</SourceControlFiles>
25      </Filter>
26      <Filter Include="Generated Files\Debug">
27        <UniqueIdentifier>{212ba673-2da8-4161-91d4-50d999f74762}</
              UniqueIdentifier>
28        <Extensions>cpp;moc</Extensions>
29        <SourceControlFiles>False</SourceControlFiles>
30      </Filter>
31      <Filter Include="Generated Files\Release">
32        <UniqueIdentifier>{95541002-1195-4132-bf15-51c12692a511}</
              UniqueIdentifier>
33        <Extensions>cpp;moc</Extensions>
34        <SourceControlFiles>False</SourceControlFiles>
35      </Filter>
36    </ItemGroup>
37    <ItemGroup>
38      <ClCompile Include="GeneratedFiles\Debug\moc_dialog_progresso.cpp">
39        <Filter>Generated Files\Debug</Filter>
40      </ClCompile>
41      <ClCompile Include="GeneratedFiles\Release\moc_dialog_progresso.cpp">
42        <Filter>Generated Files\Release</Filter>
43      </ClCompile>
44      <ClCompile Include="GeneratedFiles\Debug\moc_thread_controller.cpp">
45        <Filter>Generated Files\Debug</Filter>
46      </ClCompile>
47      <ClCompile Include="GeneratedFiles\Release\moc_thread_controller.cpp">
48        <Filter>Generated Files\Release</Filter>
49      </ClCompile>
50      <ClCompile Include="GeneratedFiles\Debug\moc_thread_worker.cpp">
51        <Filter>Generated Files\Debug</Filter>
52      </ClCompile>
53      <ClCompile Include="GeneratedFiles\Release\moc_thread_worker.cpp">
54        <Filter>Generated Files\Release</Filter>
55      </ClCompile>
56      <ClCompile Include="GeneratedFiles\Debug\moc_widget_progresso.cpp">
57        <Filter>Generated Files\Debug</Filter>
58      </ClCompile>
59      <ClCompile Include="GeneratedFiles\Release\moc_widget_progresso.cpp">
60        <Filter>Generated Files\Release</Filter>
61      </ClCompile>
62      <ClCompile Include="thread_controller.cpp">
63        <Filter>Source Files</Filter>
64      </ClCompile>
65      <ClCompile Include="GeneratedFiles\Debug\moc_thread_controller_dialog.
            cpp">
66        <Filter>Generated Files\Debug</Filter>
67      </ClCompile>
68      <ClCompile Include="GeneratedFiles\Release\moc_thread_controller_dialog
            .cpp">
69        <Filter>Generated Files\Release</Filter>
70      </ClCompile>
71      <ClCompile Include="thread_controller_dialog.cpp">
72        <Filter>Source Files</Filter>
73      </ClCompile>
74      <ClCompile Include="utilidades.cpp">
75        <Filter>Source Files</Filter>
76      </ClCompile>
77    </ItemGroup>
78    <ItemGroup>
79      <ClInclude Include="IObserver.h">
80        <Filter>Header Files</Filter>
81      </ClInclude>
82      <ClInclude Include="utilidades.h">
83        <Filter>Header Files</Filter>
84      </ClInclude>
85    </ItemGroup>
86    <ItemGroup>
87      <CustomBuild Include="dialog_progresso.h">
88        <Filter>Header Files</Filter>
89      </CustomBuild>
90      <CustomBuild Include="thread_controller.h">
91        <Filter>Header Files</Filter>
92      </CustomBuild>
93      <CustomBuild Include="thread_worker.h">
94        <Filter>Header Files</Filter>
95      </CustomBuild>
```

```
96      <CustomBuild Include="widget_progresso.h">
97        <Filter>Header Files</Filter>
98      </CustomBuild>
99      <CustomBuild Include="thread_controller_dialog.h">
100       <Filter>Header Files</Filter>
101     </CustomBuild>
102   </ItemGroup>
103 </Project>
```

Listagem B.163: lib_utilidades.vcxproj.filters

```
1  #pragma once
2
3  #include <QDialog>
4
5  class DialogProgresso : public QDialog
6  {
7  public:
8      DialogProgresso(QWidget *parent = 0) : QDialog(parent){};
9
10     Q_OBJECT
11     public slots:
12         virtual void onUpdateProgresso(int *porcentagem) = 0;
13     signals:
14         void pararExecucaoThread();
15 };
```

Listagem B.164: dialog_progresso.h

```
1  #ifndef WIDGETPROGRESSO_H
2  #define WIDGETPROGRESSO_H
3
4  #include <QWidget>
5
6  class WidgetProgresso : public QWidget
7  {
8      Q_OBJECT
9
10     public:
11         WidgetProgresso(QWidget *parent = 0) : QWidget(parent) {};
12
13     public slots:
14         virtual void onUpdateProgresso(int *porcentagem) = 0;
15     signals:
16         void pararExecucaoThread();
17 };
18 #endif
```

Listagem B.165: widget_progresso.h

```
1  #pragma once
2
3  class IObserver
4  {
5  public:
6      virtual void Notificar(int *porcentagem) = 0;
7  };
```

Listagem B.166: IObserver.h

```
1  #pragma once
2
3  #include "IObserver.h"
4  #include <QThread>
5
6  class ThreadWorker : public QObject, public IObserver
7  {
8      Q_OBJECT
9      public slots:
10         // função que deve ser implementada
```

```
11          // para que o código seja executado em outra thread
12          virtual void executarCodigo() = 0;
13          virtual void pararExecucao() = 0;
14      signals:
15          void updateProgresso(int *porcentagem);
16          void finalizouExecucao();
17  };
```

Listagem B.167: thread_worker.h

```
1   #include "thread_controller.h"
2
3   ThreadController::ThreadController() :
4       m_worker(0), m_thread(0)
5   {
6   }
7
8   ThreadController::~ThreadController()
9   {
10      liberaMemoria();
11  }
12
13  void ThreadController::setarWorker(ThreadWorker* worker)
14  {
15      if (m_worker) delete m_worker;
16
17      m_worker = worker;
18  }
19
20  void ThreadController::iniciarExecucaoWorker(WidgetProgresso* telaPai)
21  {
22      if (m_thread)
23      {
24          delete m_thread;
25      }
26      m_thread = new QThread();
27
28      if (telaPai)
29      {
30          QObject::connect(m_worker, SIGNAL(updateProgresso(int*)), telaPai,
                SLOT(onUpdateProgresso(int*)));
31          //QObject::connect(telaPai, SIGNAL(pararExecucaoThread()), m_worker
                , SLOT(pararExecucao()));
32      }
33
34      QObject::connect(m_worker, SIGNAL(finalizouExecucao()), this, SLOT(
            workerFinalizou()));
35      QObject::connect(m_thread, SIGNAL(started()), m_worker, SLOT(
            executarCodigo()));
36
37
38      m_worker->moveToThread(m_thread);
39      m_thread->start();
40  }
41
42  void ThreadController::workerFinalizou()
43  {
44      emit execucaoFinalizada();
45      emit execucaoFinalizada(this);
46  }
47
48  void ThreadController::finalizarExecucao()
49  {
50      liberaMemoria();
51  }
52
53  void ThreadController::liberaMemoria()
54  {
55      if (m_worker) delete m_worker;
56      if (m_thread) delete m_thread;
57  }
```

Listagem B.168: thread_controller.cpp

```
1   #pragma once
2
```

```
3  #include <QThread>
4  #include <QObject>
5  #include "thread_worker.h"
6  #include "widget_progresso.h"
7
8  class ThreadController : public QObject
9  {
10     Q_OBJECT
11
12     public:
13         ThreadController();
14         ~ThreadController();
15         // configura qual thread vai ser rodada
16         void setarWorker(ThreadWorker* worker);
17         ThreadWorker* getThreadWorker(){return m_worker;}
18         // inicia a execução do código da thread worker
19         void iniciarExecucaoWorker(WidgetProgresso* telaPai);
20         // finaliza a thread e limpa a memória
21         void finalizarExecucao();
22     private:
23         QThread* m_thread;
24         ThreadWorker* m_worker;
25
26         // limpa a memória
27         void liberaMemoria();
28     public slots:
29         void workerFinalizou();
30
31     signals:
32         void execucaoFinalizada();
33         void execucaoFinalizada(ThreadController*);
34 };
```

Listagem B.169: thread_controller.h

```
1  #include "thread_controller_dialog.h"
2
3  ThreadControllerDialog::ThreadControllerDialog() :
4      m_worker(0), m_thread(0)
5  {
6  }
7
8  ThreadControllerDialog::~ThreadControllerDialog()
9  {
10     liberaMemoria();
11 }
12
13 void ThreadControllerDialog::setarWorker(ThreadWorker* worker)
14 {
15     if (m_worker) delete m_worker;
16
17     m_worker = worker;
18 }
19
20 void ThreadControllerDialog::iniciarExecucaoWorker(DialogProgresso* telaPai
       )
21 {
22     if (m_thread) delete m_thread;
23
24     m_thread = new QThread();
25     QObject::connect(m_worker, SIGNAL(updateProgresso(int*)), telaPai, SLOT
           (onUpdateProgresso(int*)));
26     QObject::connect(m_worker, SIGNAL(finalizouExecucao()), this, SLOT(
           workerFinalizou()));
27     QObject::connect(m_thread, SIGNAL(started()), m_worker, SLOT(
           executarCodigo()));
28     //QObject::connect(telaPai, SIGNAL(pararExecucaoThread()), m_worker,
           SLOT(pararExecucao()));
29
30
31     m_worker->moveToThread(m_thread);
32     m_thread->start();
33 }
34
35 void ThreadControllerDialog::workerFinalizou()
36 {
37     emit execucaoFinalizada();
38 }
39
```

```
40 void ThreadControllerDialog::finalizarExecucao()
41 {
42     liberaMemoria();
43 }
44
45 void ThreadControllerDialog::liberaMemoria()
46 {
47     if (m_worker)
48     {
49         delete m_worker;
50         m_worker = NULL;
51     }
52     if (m_thread)
53     {
54         delete m_thread;
55         m_thread = NULL;
56     }
57 }
```

Listagem B.170: thread_controller_dialog.cpp

```
1  #pragma once
2
3  #include <QThread>
4  #include <QObject>
5  #include "thread_worker.h"
6  #include "dialog_progresso.h"
7
8  class ThreadControllerDialog : public QObject
9  {
10     Q_OBJECT
11
12     public:
13         ThreadControllerDialog();
14         ~ThreadControllerDialog();
15         // configura qual thread vai ser rodada
16         void setarWorker(ThreadWorker* worker);
17         ThreadWorker* getThreadWorker(){return m_worker;}
18         // inicia a execução do código da thread worker
19         void iniciarExecucaoWorker(DialogProgresso* telaPai);
20         // finaliza a thread e limpa a memória
21         void finalizarExecucao();
22     private:
23         QThread* m_thread;
24         ThreadWorker* m_worker;
25
26         // limpa a memória
27         void liberaMemoria();
28     public slots:
29         void workerFinalizou();
30
31     signals:
32         void execucaoFinalizada();
33 };
```

Listagem B.171: thread_controller_dialog.h

```
1  #include "utilidades.h"
2
3  #include <math.h>
4  #include <sstream>
5  #include <iostream>
6  #include <fstream>
7  #include <iomanip>
8  #include <regex>
9  #include <algorithm>
10
11 #include <Qlocale>
12 #include <QCoreApplication>
13
14 const string Utilidades::coresDistintas[] = {"#FF0000", "#00FF00", "#0000FF
       ", "#FFFF00", "#FF00FF", "#00FFFF", "#000000",
15         "#800000", "#008000", "#000080", "#808000", "#800080", "#008080", "
               #808080",
16         "#C00000", "#00C000", "#0000C0", "#C0C000", "#C000C0", "#00C0C0", "
               #C0C0C0",
```

```
17          "#400000", "#004000", "#000040", "#404000", "#400040", "#004040", "
                #404040",
18          "#200000", "#002000", "#000020", "#202000", "#200020", "#002020", "
                #202020",
19          "#600000", "#006000", "#000060", "#606000", "#600060", "#006060", "
                #606060",
20          "#A00000", "#00A000", "#0000A0", "#A0A000", "#A000A0", "#00A0A0", "
                #A0A0A0",
21          "#E00000", "#00E000", "#0000E0", "#E0E000", "#E000E0", "#00E0E0", "
                #E0E0E0"};
22
23 string Utilidades::dblToStdStr(const double valor, const int precisao,
       const bool separadorDecimalVirgula, const bool omitirSeparadorMilhar)
24 {
25     QLocale locale = QLocale(QLocale::Portuguese, QLocale::Brazil);
26
27     if (!separadorDecimalVirgula)
28     {
29         //ponto como separador
30         locale = QLocale(QLocale::English, QLocale::UnitedStates);
31     }
32
33     if (omitirSeparadorMilhar)
34     {
35         locale.setNumberOptions(QLocale::OmitGroupSeparator);
36     }
37
38     QString resultado = locale.toString(valor, 'f', precisao);
39
40     if (resultado == "nan") {return "Formato inválido";}
41
42     QChar zeroChar = locale.toString(0).at(0);
43
44     if (precisao > 0)
45     {
46         while (resultado.size() > 1 && resultado[resultado.size() - 1] ==
                zeroChar)
47         {
48             resultado.remove(resultado.size() - 1, 1);
49         }
50
51         if (resultado[resultado.size() - 1] == locale.decimalPoint())
52         {
53             resultado.remove(resultado.size() - 1, 1);
54         }
55     }
56
57     return resultado.toLatin1().data();
58 }
59
60 double Utilidades::stdStrToDbl(const string &strStd, const bool
       separadorDecimalVirgula, const bool omitirSeparadorMilhar, bool *ok)
61 {
62     if (ok) {*ok = true;}
63
64     QString strQt = strStd.c_str();
65
66     bool okQt;
67
68     QLocale locale = separadorDecimalVirgula ?
69         QLocale(QLocale::Portuguese, QLocale::Brazil)
70         : locale = QLocale(QLocale::English, QLocale::UnitedStates);
71
72     if (omitirSeparadorMilhar)
73     {
74         locale.setNumberOptions(QLocale::OmitGroupSeparator);
75     }
76
77     double valueDbl = locale.toDouble(strQt, &okQt);
78
79     if (ok) {*ok = okQt;}
80
81     if (okQt)
82     {
83         return  valueDbl;
84     }
85
86     return numeric_limits<double>::quiet_NaN();
87 }
88
89 vector<double> Utilidades::stdStrToDbl(const vector<string> &strs, const
       bool separadorDecimalVirgula, const bool omitirSeparadorMilhar, vector
```

```
        <size_t> &idxNaoOk)
90  {
91      vector<double> dbls = vector<double>();
92
93      bool ok;
94
95      for (vector<string>::const_iterator citStr = strs.cbegin();
96          citStr != strs.end();
97          citStr++)
98      {
99          dbls.push_back(stdStrToDbl(*citStr, separadorDecimalVirgula,
                omitirSeparadorMilhar, &ok));
100         if (!ok)
101         {
102             idxNaoOk.push_back(citStr - strs.cbegin());
103         }
104     }
105
106     return dbls;
107 }
108
109 vector<string> Utilidades::splitStr(const string &str, const string &
        delimitador, bool considerarVazios)
110 {
111     string copyStr = str;
112     vector<string> tokens;
113
114     size_t pos = 0;
115     string token;
116     while ((pos = copyStr.find(delimitador)) != string::npos)
117     {
118         token = copyStr.substr(0, pos);
119         if (considerarVazios || (!considerarVazios && !tokens.empty()))
120         {
121             tokens.push_back(token);
122         }
123         copyStr.erase(0, pos + delimitador.length());
124     }
125
126     if (considerarVazios || (!considerarVazios && !copyStr.empty()))
127     {
128         tokens.push_back(copyStr);
129     }
130     return tokens;
131 }
132
133 string Utilidades::mergeStr(const vector<string> &strs, const string &
        delimitador)
134 {
135     string result = "";
136
137     for (int i = 0; i < strs.size() - 1; i++)
138     {
139         result += strs[i] + delimitador;
140     }
141
142     result += strs.back();
143
144     return result;
145 }
146
147 double Utilidades::arrendondar(const double valor, const int precisao)
148 {
149     double fatorDeslocamento = std::pow(10.0, precisao);
150     return floor((valor * fatorDeslocamento) + 0.5) / fatorDeslocamento;
151 }
152
153 string Utilidades::dblToMoedaCorrente(const double valor, const bool dollar
        )
154 {
155     QLocale locale = QLocale(QLocale::Portuguese, QLocale::Brazil);
156
157     if (dollar)
158     {
159         locale = QLocale(QLocale::English, QLocale::UnitedStates);
160     }
161
162     return locale.toCurrencyString(valor).toStdString();
163 }
164
165 bool Utilidades::is_integer(const double valor)
166 {
```

```
167     double intpart, fractpart;
168     fractpart = modf(valor, &intpart);
169 
170     return fractpart == 0.0;
171 }
172 
173 bool Utilidades::parseCVSDados(const string &delimitador, const string &
        arquivo, vector<vector<string>> &dadosRetorno, vector<string> &
        nomesRetorno, QProgressDialog *progress)
174 {
175     try
176     {
177 
178         const double tamanhoArquivoBytes = file_size(arquivo);
179         double totalLidoBytes = 0;
180         int tamanhoParteArquivo = tamanhoArquivoBytes / 100;
181         if (tamanhoParteArquivo <= 0)
182         {
183             tamanhoParteArquivo = 1;
184         }
185 
186         ifstream arquivo(arquivo.c_str());
187         if (!arquivo.good())
188         {
189             arquivo.close();
190             return false;
191         }
192 
193         string linha;
194         getline(arquivo, linha);
195 
196         totalLidoBytes += linha.size() + 2; //1 byte por char e +2 pelo \n\
                r
197 
198         nomesRetorno = splitStr(linha, delimitador);
199 
200         int ultimaParte = 0;
201 
202         while(getline(arquivo, linha))
203         {
204             vector<string> linhaVetor = splitStr(linha, delimitador);
205             dadosRetorno.push_back(linhaVetor);
206 
207             totalLidoBytes += linha.size() + 2; //1 byte por char e +2 pelo
                    \n\r
208 
209             int parteAtual = (int)totalLidoBytes / tamanhoParteArquivo;
210 
211             if (progress && tamanhoArquivoBytes > 0 && (parteAtual >
                    ultimaParte)) //atualiza a cada 1% do arquivo lido
212             {
213                 progress->setValue(min((int)(totalLidoBytes /
                        tamanhoArquivoBytes * 50), 99));
214                 QCoreApplication::processEvents(QEventLoop::
                        ExcludeUserInputEvents);
215                 ultimaParte = parteAtual;
216             }
217         }
218 
219         arquivo.close();
220         return true;
221     }
222     catch(...)
223     {
224         return false;
225     }
226 
227     return false;
228 }
229 
230 long Utilidades::file_size(const string &dir)
231 {
232     long size = -1;
233 
234     try
235     {
236         streampos begin, end;
237         ifstream myfile (dir, ios::binary);
238         begin = myfile.tellg();
239         myfile.seekg (0, ios::end);
240         end = myfile.tellg();
241         myfile.close();
```

```
242         size = end - begin;
243     }
244     catch (...) {}
245 
246     return size;
247 }
```

Listagem B.172: utilidades.cpp

```
 1 #pragma once
 2 
 3 #include <string>
 4 #include <vector>
 5 #include <QProgressDialog>
 6 
 7 using namespace std;
 8 
 9 class Utilidades
10 {
11 public:
12     static const string coresDistintas[];
13     static string Utilidades::dblToStdStr(const double valor, const int
           precisao = 2, const bool separadorDecimalVirgula = true, const
           bool omitirSeparadorMilhar = true);
14     static double stdStrToDbl(const string &str, const bool
           separadorDecimalVirgula, const bool omitirSeparadorMilhar, bool *
           ok = NULL);
15     static vector<double> stdStrToDbl(const vector<string> &strs, const
           bool separadorDecimalVirgula, const bool omitirSeparadorMilhar,
           vector<size_t> &idxNaoOk);
16     static vector<string> splitStr(const string &str, const string &
           delimitador, bool considerarVazios = true);
17     static string mergeStr(const vector<string> &strs, const string &
           delimitador);
18     static double arrendondar(const double valor, const int precisao);
19     static string dblToMoedaCorrente(const double valor, const bool dollar)
           ;
20     static bool is_integer(const double x);
21     static bool parseCVSDados(const string &delimitador, const string &
           arquivo, vector<vector<string>> &dadosRetorno, vector<string> &
           nomesRetorno, QProgressDialog *progress = NULL);
22     static long file_size(const string &dir);
23 };
24 
25 template <typename T>
26 bool comparePointedValueLT(const T *a, const T *b)
27 {
28    return (*a < *b);
29 }
30 
31 template <typename T>
32 bool comparePointedValueEQ(const T *a, const T *b)
33 {
34    return (*a == *b);
35 }
36 
37 #ifndef isnan
38 inline bool isnan(const double x)
39 {
40     return x != x;
41 }
42 #endif
```

Listagem B.173: utilidades.h