

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

PROJETO ENGENHARIA
DESENVOLVIMENTO DO SOFTWARE
CÁLCULO DO ÍNDICE DE PRODUTIVIDADE DE POÇOS
HORIZONTAIS E VERTICAIS
TRABALHO DA DISCIPLINA PROGRAMAÇÃO PRÁTICA

Versão 1:
CAROLINA BASTOS E DOUGLAS RIBEIRO
Prof. André Duarte Bueno

MACAÉ - RJ
Agosto - 2021

Sumário

1	Introdução	1
1.1	Escopo do problema	1
1.2	Objetivos	1
2	Especificação	3
2.1	O que é a especificação?	3
2.2	Nome do sistema/produto	3
2.3	Especificação	3
2.3.1	Requisitos funcionais	4
2.3.2	Requisitos não funcionais	4
2.4	Casos de uso	5
2.4.1	Diagrama de caso de uso geral	5
2.4.2	Diagrama de caso de uso específico	5
3	Elaboração	7
3.1	Análise de domínio	7
3.2	Formulação teórica	8
3.2.1	Produtividade de Poços	8
3.2.2	Índice de Produtividade	8
3.2.3	Efeito Skin	8
3.2.4	Regime Permanente	9
3.2.5	Produtividade de Poços Horizontais	9
3.2.6	Cálculo do IP com anisotropia	10
3.3	Diagrama de pacotes – assuntos	11
4	AOO – Análise Orientada a Objeto	13
4.1	Diagramas de classes	13
4.1.1	Dicionário de classes	15
4.2	Diagrama de seqüência – eventos e mensagens	15
4.2.1	Diagrama de seqüência geral	15
4.3	Diagrama de comunicação – colaboração	16
4.4	Diagrama de máquina de estado	17

4.5	Diagrama de atividades	18
5	Projeto	19
5.1	Projeto do sistema	19
5.2	Projeto orientado a objeto – POO	20
5.3	Diagrama de componentes	24
5.4	Diagrama de implantação	25
6	Implementação	26
6.1	Código fonte	26

Capítulo 1

Introdução

No presente projeto de engenharia desenvolve-se o software Cálculo do Índice de Produtividade de Poços Horizontais e Verticais, um software aplicado a engenharia de petróleo e que utiliza o paradigma da orientação a objetos.

O software é da área de engenharia de poços e permite simular a influência de parâmetros do reservatório e do fluido na produtividade dos poços, podendo os resultados serem comparados para definir qual melhor design de poço para cada cenário.

1.1 Escopo do problema

O projeto de perfuração de um poço horizontal é diferente de um projeto de perfuração de um poço vertical porque a produtividade do poço depende do comprimento do mesmo, além de fatores determinantes em ambos os projetos como permeabilidade, anisotropias, espessura permeável, viscosidade do óleo e vários aspectos relativos à perfuração do trecho horizontal. Para cada diferente cenário, haverá uma diferente solução de poço para desenvolver o campo. E isto engloba além do fator financeiro, a capacidade produtiva desses poços, se será ou não vantajoso a exploração do mesmo.

Por isso, é importante evidenciar em que situações, em termos de produtividade, qual design de poço seria mais recomendado através de uma comparação de resultados, alinhado com um embasamento teórico, sobre diversos parâmetros de reservatório que podem intervir na produtividade do poço horizontal e qual o ganho de produtividade em relação ao poço vertical. Assim os engenheiros de reservatório e de poço podem trabalhar de forma conjunta para escolher a técnica mais apropriada.

1.2 Objetivos

Os objetivos deste projeto de engenharia são:

- Objetivo geral:

- Desenvolver um projeto de engenharia de software para resolver os diferentes modelos matemáticos de previsão de produtividade de poços horizontais e verticais e a influência dos parâmetros de reservatórios nos mesmos para analisar em que situações, em termos de produtividade, qual design de poço seria mais recomendado através das simulações.
- Objetivos específicos:
 - Modelar física e matematicamente o problema.
 - Modelagem estática do software (diagramas de caso de uso, de pacotes, de classes).
 - Modelagem dinâmica do software (desenvolver algoritmos e diagramas exemplificando o fluxo de processamento).
 - Calcular o Índice de Produtividade (IP) dos poços, a partir dos modelos analíticos de Borisov, Giger, Joshi e RenardDupuy e Shedid.
 - Simular a influência de parâmetros do reservatório, como a altura, a anisotropia, a centralização vertical e a viscosidade do fluido nos resultados do IP.
 - Implementar manual simplificado de uso do software.

Capítulo 2

Especificação

Apresenta-se neste capítulo do projeto de engenharia a concepção, a especificação do sistema a ser modelado e desenvolvido.

2.1 O que é a especificação?

Nesta seção são descritas as principais características, além dos requisitos para a utilização do software desenvolvido.

2.2 Nome do sistema/produto

Nome	Cáculo do Índice de Produtividade de Poços Horizontais e Verticais
Componentes principais	Sistema para calcular a influência das propriedades do reservatório e do fluido na produtividade dos poços horizontais e verticais a fim de definir qual melhor design para o poço.
Missão	Simular diferentes cenários do sistema fluido/reservatório e sua influência na produtividade dos poços. Calcular IP dos poços. Gerar gráficos que permita comparar IP de poços com diferentes desigs (horizontal/vertical).

2.3 Especificação

Apresenta-se a seguir a especificação do software.

O projeto a ser desenvolvido consiste de um programa que deverá realizar cálculos de IP de poços horizontais e verticais, além de mostrar os resultados graficamente. Os cálculos serão feitos a partir de modelos matemáticos existentes na literatura e na dinâmica de execução do software, o usuário poderá escolher qual modelo deseja utilizar, qual o tipo de formação a ser atravessada e as características do fluido produzido. Além disso, o usuário deverá entrar com os dados do reservatório (permeabilidades horizontal e vertical, espessura, comprimento e raio do poço) e viscosidade e fator de formação do fluido - ou poderá inserir esses dados em um arquivo .txt e usá-lo como input do programa. Ao final da simulação o usuário poderá ver os resultados em tela, gerar gráficos e salvá-los como imagem.

2.3.1 Requisitos funcionais

Apresenta-se a seguir os requisitos funcionais.

RF-01	O programa deverá solicitar os dados de entrada (parâmetros do poço, do reservatório e do fluido) ao usuário.
RF-02	O programa também deverá permitir o carregamento desses dados de entrada a partir de um arquivo de disco criado pelo usuário.
RF-03	O usuário deverá ter liberdade para escolher o tipo de formação que o poço irá atravessar (isotrópica ou anisotrópica)
RF-04	O usuário deverá ter liberdade para escolher o modelo matemático para o cálculo do IP.
RF-05	O programa deverá mostrar os resultados dos cálculos de IP na tela.
RF-06	O usuário poderá plotar seus resultados em um gráfico, podendo este ser salvo como imagem.

2.3.2 Requisitos não funcionais

RNF-01	Os cálculos devem ser feitos utilizando-se formulações/modelos matemáticos conhecidos na literatura.
RNF-02	O programa deverá ser multi-plataforma, podendo ser executado em <i>Windows</i> , <i>GNU/Linux</i> ou <i>Mac</i> .

2.4 Casos de uso

A tabela 2.1 apresenta um caso de uso do sistema.

Tabela 2.1: Exemplo de caso de uso

Nome do caso de uso:	Cálculo do IP de um poço horizontal
Resumo/descrição:	Determinar a capacidade produtiva de um poço do tipo horizontal, a partir de um modelo matemático a ser escolhido
Etapas:	<ol style="list-style-type: none"> 1. Entrar com os dados do poço, do reservatório e do fluido (permeabilidade, espessura, viscosidade, etc.). 2. Definir o tipo de formação a ser atravessada pelo poço: isotrópica ou anisotrópica. 3. Definir o modelo matemático mais apropriado para aquele cenário de reservatório/fluido a partir da análise dos resultados. 4. Salvar resultados em disco.
Cenários alternativos:	Inserir valores negativos para parâmetros do reservatório ou incompatíveis com a ordem de grandeza do problema real.

2.4.1 Diagrama de caso de uso geral

O diagrama de caso de uso geral da Figura 2.1 mostra o usuário interagindo com o software para obter o IP de um poço. Neste caso de uso geral, o usuário insere os dados de entrada (via tela ou através de um arquivo .txt) para então analisar o resultado obtido.

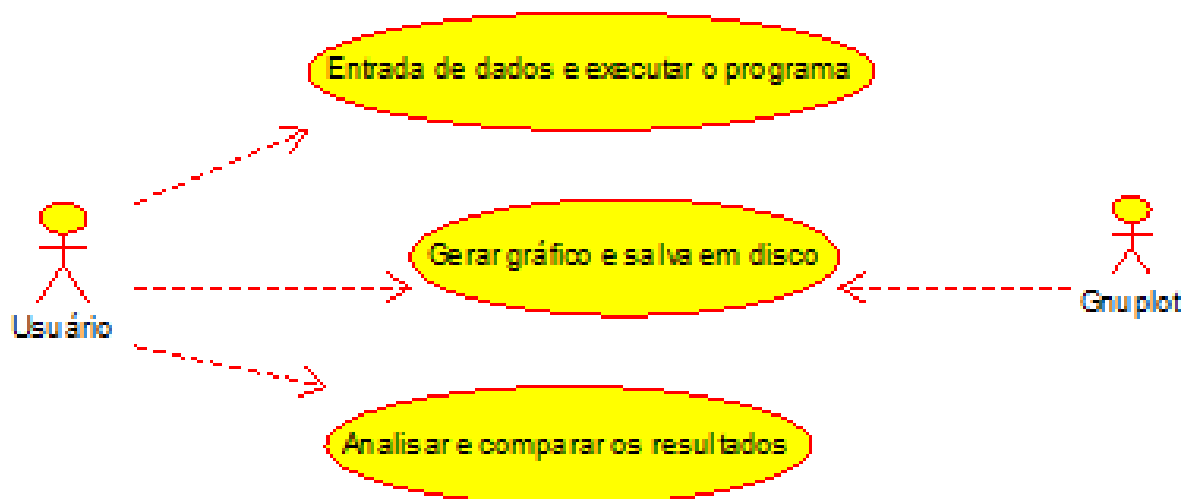


Figura 2.1: Diagrama de caso de uso geral – Cálculo do IP

2.4.2 Diagrama de caso de uso específico

O caso de uso Comparar IP de um reservatório anisotrópico e isotrópico descrito na Figura 2.1 e na Tabela 2.1 é detalhado na Figura 2.2. O usuário pode variar os parâmetros

do poço e do reservatório e então plotar esses diferentes cenários em um gráfico para fazer comparações e definir qual melhor se adequa ao projeto.

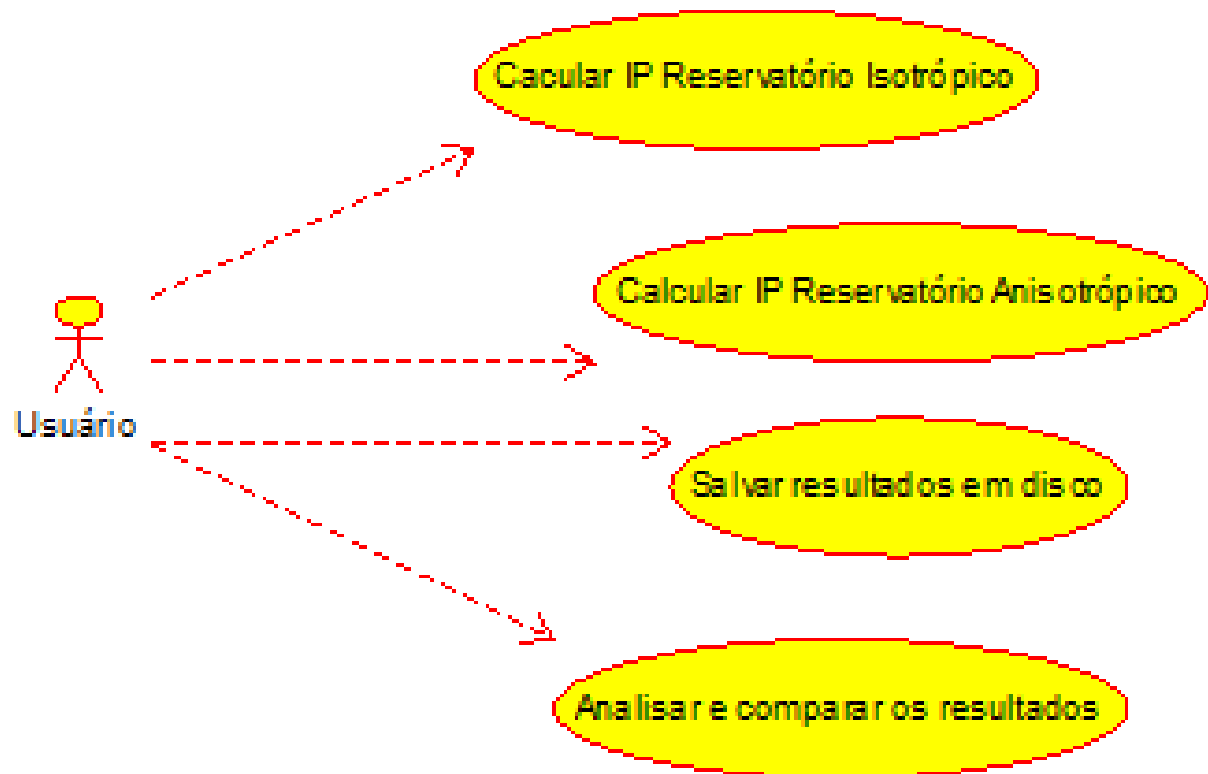


Figura 2.2: Diagrama de caso de uso específico – Comparando o IP de reservatório isotrópico com um anisotrópico

Capítulo 3

Elaboração

Depois da definição dos objetivos, da especificação do software e da montagem dos primeiros diagramas de caso de uso, a equipe de desenvolvimento do projeto de engenharia passa por um processo de elaboração que envolve o estudo de conceitos relacionados ao sistema a ser desenvolvido, a análise de domínio e a identificação de pacotes.

Na elaboração fazemos uma análise dos requisitos, ajustando os requisitos iniciais de forma a desenvolver um sistema útil, que atenda às necessidades do usuário e, na medida do possível, permita seu reuso e futura extensão.

Eliminam-se os requisitos "impossíveis" e ajusta-se a idéia do sistema de forma que este seja flexível, considerando-se aspectos como custos e prazos.

3.1 Análise de domínio

A tecnologia de poços horizontais constitui o padrão de perfuração e implementação de poços de desenvolvimento na indústria do petróleo, ao lado da perfuração direcional, principalmente em ambientes offshore, devido ao alto custo de um poço. Antes do avanço da tecnologia para a perfuração de poços horizontais a desvantagem em relação a poços verticais era que apenas uma área poderia ser drenada por um mesmo poço.

A partir do surgimento de novas técnicas de perfuração passaram-se a perfurar poços horizontais multilaterais, assim um poço poderia drenar mais de um reservatório. A partir de um poço vertical perfuram-se vários trechos horizontais em diferentes camadas. O principal motivo para esse tipo é o grande aumento que se dá de produtividade, podendo apontar outras vantagens em relação ao poço vertical como menor gradiente de pressão, menor número de poços, maior exposição ao reservatório, poços de longo alcance, redução da produção de areia, entre outros. Porém, como qualquer outro método há desvantagens, por exemplo, se o poço horizontal for atingido pela água proveniente do contato óleo/água ascendente, dependendo da completação que foi utilizada no poço, ele deverá ser fechado ou transformado em um poço injetor, não podendo haver intervenção ou recompletação.

O projeto de perfuração de um poço horizontal é diferente de um poço vertical, porque

a sua produtividade depende de seu comprimento, além de fatores determinantes em ambos os projetos como viscosidade do óleo e permeabilidade da formação e vários aspectos relativos à perfuração do trecho horizontal.

Este projeto tem como objetivo evidenciar em que situações, em termos de produtividade, qual design de poço seria mais recomendado por meio de um estudo com embasamento teórico sobre diversos parâmetros de reservatório que podem intervir na produtividade do poço horizontal e qual o ganho de produtividade em relação a um vertical.

Depois de estudar as especificações do sistema e estudos de biblioteca e de disciplinas do curso foi possível identificar nosso domínio de trabalho:

- O software irá calcular vários índices de produtividade para um mesmo reservatório dado por meio dos métodos a depender do caso ser isotrópico ou anisotrópico;
- O software usará conceitos de engenharia de reservatório e da engenharia de poço para que se realize as simulações, aqui iremos ter explicações básicas de quando se usar cada método, porém é necessário que se tenha o conhecimento básico dessas disciplinas para a realização da simulação.
- O software plotará os resultados dos índices de produtividade para poços com diferentes design.

3.2 Formulação teórica

3.2.1 Produtividade de Poços

Inicialmente, serão apresentadas algumas definições de parâmetros para uma boa compreensão de termos e conceitos utilizados no decorrer do projeto.

3.2.2 Índice de Produtividade

O índice de produtividade, de forma simplificada, é dado pela equação 3.1 :

$$IP = \frac{Q}{P_e - P_w} \quad (3.1)$$

Onde:

$Q = \text{vazão} [m^3/d]$

$P_e = \text{pressão estática do reservatório} [kgf/cm^2]$

$P_w = \text{pressão de fluxo do poço} [kgf/cm^2]$

3.2.3 Efeito Skin

Segundo [JOSHI, 1988] o efeito de película ou de skin é um modelo matemático introduzido na indústria de petróleo por Van Everdingen & Hurst com o objetivo de simular

um fenômeno real, o dano à formação.

A partir da definição do fator de skin pode-se definir o raio efetivo do poço por meio da equação 3.2:

$$r'_w = r e^{-s} \quad (3.2)$$

Onde:

r'_w = raio efetivo do poço [cm]

r_w = raio do poço [cm]

s = fator de skin

3.2.4 Regime Permanente

As soluções analíticas em estado estacionário ou permanente são a forma mais simples de soluções para poços horizontais. No regime de fluxo permanente, por hipótese admitimos que a pressão em qualquer ponto do reservatório é independente do tempo.

Na realidade são pouquíssimos casos de reservatórios que operam sob as condições do regime de fluxo permanente. Apesar disso, essas soluções são usadas em grande frequência segundo [JOSHI, 1988] pelos seguintes fatos:

1. São de fácil dedução analítica;
2. Podem ser usados para obter soluções para o fluxo transiente, usa-se o artifício de aumentar o raio de drenagem com o tempo;
3. Podem ser usadas para se obter soluções para o fluxo pseudopermanente por meio do emprego do fator de Dietz, que permite o cálculo da pseudopressão para diversas geometrias do reservatório;
4. Podem ser verificadas experimentalmente por meio de modelos de laboratório [ROSA, 2006].

3.2.5 Produtividade de Poços Horizontais

Os métodos abaixo são para formações isotrópicas, ou seja, com a permeabilidade vertical igual à horizontal.

- Borisov:

$$IP = \frac{\frac{2\pi k_h h}{\mu}}{\ln\left(\frac{4r_{eh}}{L}\right) + \left[\left(\frac{h}{L}\right)\ln\left(\frac{h}{2\pi r_w}\right)\right]} \quad (3.3)$$

- Giger:

$$IP = \frac{\frac{2\pi k_h h}{\mu}}{\left(\frac{L}{h}\right)\ln\left(\frac{1+\sqrt{1-\left(\frac{L}{2r_{eh}}\right)^2}}{\frac{L}{2r_{eh}}}\right) + \ln\left(\frac{h}{2\pi r_w}\right)} \quad (3.4)$$

- Joshi:

$$IP = \frac{2\pi k_h h}{\ln\left(\frac{a + \sqrt{a^2 - (\frac{L}{2})^2}}{\frac{L}{2}}\right) + \left(\frac{h}{L}\right)\ln\left(\frac{h}{2r_w}\right)} \quad (3.5)$$

Onde:

$$a = \left(\frac{L}{2}\right)\sqrt{0.5 + \sqrt{0.25 + \left(\frac{2r_{eh}}{L}\right)^4}} \quad (3.6)$$

IP = Índice de Produtividade

k_h = permeabilidade horizontal

h = altura do reservatório

μ = viscosidade do óleo

r_{eh} = raio exeterno do reservatório

L = comprimento horizontal do reservatório

r_w = raio do poço

Na literatura também é apresentado uma solução que é independente do raio de drenagem r_{eh} do poço. segundo [SHEDID, 2001]:

$$IP = \frac{\frac{2\pi h k}{\mu B_o}}{\left[\ln\left(\frac{h/2r_w}{L/h}\right) + \left(0.25 + \frac{C}{L}\right)\left(\frac{1}{r_w} - \frac{2}{h}\right)\right]} \quad (3.7)$$

Onde:

B_o = fator volume de formação do óleo

E a contante C é mostrada na figura 3.1 abaixo:

Horizontal well Length (L), ft	Value of (C) or equation to be used to calculate the constant (C), ft
>0-1000	270
>1000-3000	$C = 470 - 0.20 * L$

Figura 3.1: Constante C

3.2.6 Cálculo do IP com anisotropia

$$IP = \frac{Q}{\Delta P} = \frac{\frac{2\pi k_h h}{\mu}}{\ln\left(\frac{a + \sqrt{a^2 - (\frac{L}{2})^2}}{\frac{L}{2}}\right) + \left(\frac{\beta h}{L}\right)\ln\left(\frac{\beta h}{2r_w}\right)} \quad (3.8)$$

Onde:

$$\beta = \sqrt{\frac{k_h}{k_v}} \quad (3.9)$$

ΔP é a queda de pressão no reservatório

- Modelo de Renard e Dupuy:

$$IP = \frac{2\pi k_h h}{\mu} \left(\frac{1}{\cosh^{-1}(X) + \left(\frac{\beta h}{L}\right) \left(\ln \left[\frac{h}{2\pi r'_w}\right]\right)} \right) \quad (3.10)$$

Onde:

$$r'_w = \frac{1 + \beta}{2\beta} r_w \quad (3.11)$$

$$X = \frac{2a}{L} \quad (3.12)$$

Isso é usado para uma área elipsoidal, a é dado pela equação 3.6 e β pela equação 3.9.

3.3 Diagrama de pacotes – assuntos

Com base na análise de domínio do software desenvolvido, foram identificados os seguintes pacotes:

- Pacote Fluido: Engloba as características do fluido, como viscosidade e fator volume formação.
- Pacote Reservatório: Contém os dados relativos ao reservatório, incluindo o tipo de formação, se é isotrópica ou anisotrópica.
- Pacote Poço: Contém os dados relativos ao poço e os métodos que serão utilizados para o cálculo do índice de produtividade (subsistema MetodosIP).
- Pacote Gráficos: Usando o software Gnuplot, será possível gerar gráficos relacionando cada índice de produtividade com cada método.
- Pacote Simulador: Relaciona os pacotes acima, sendo responsável pela criação e destruição dos objetos.

Veja Figura 3.2.

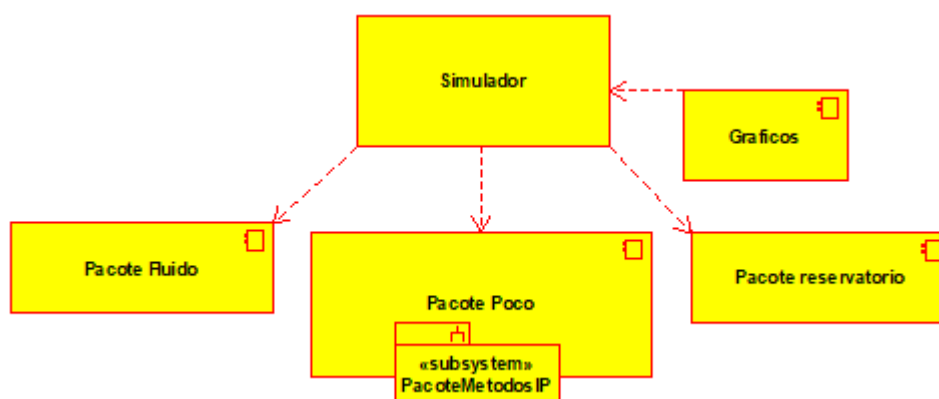


Figura 3.2: Diagrama de Pacotes

Capítulo 4

AOO – Análise Orientada a Objeto

Nesta etapa de desenvolvimento do projeto de engenharia, apresentamos a Análise Orientada a Objeto - AOO. Esta análise mostra as relações entre as classes, os atributos, os métodos e suas associações e consiste em modelos estruturais dos objetos e seus relacionamentos, e modelos dinâmicos, apresentando as modificações do objeto com o tempo. O resultado da análise é um conjunto de diagramas que identificam os objetos e seus relacionamentos.

4.1 Diagramas de classes

O diagrama de classes do software desenvolvido é apresentado na Figura 4.1. Como podemos perceber, ele é constituído de quatorze classes.

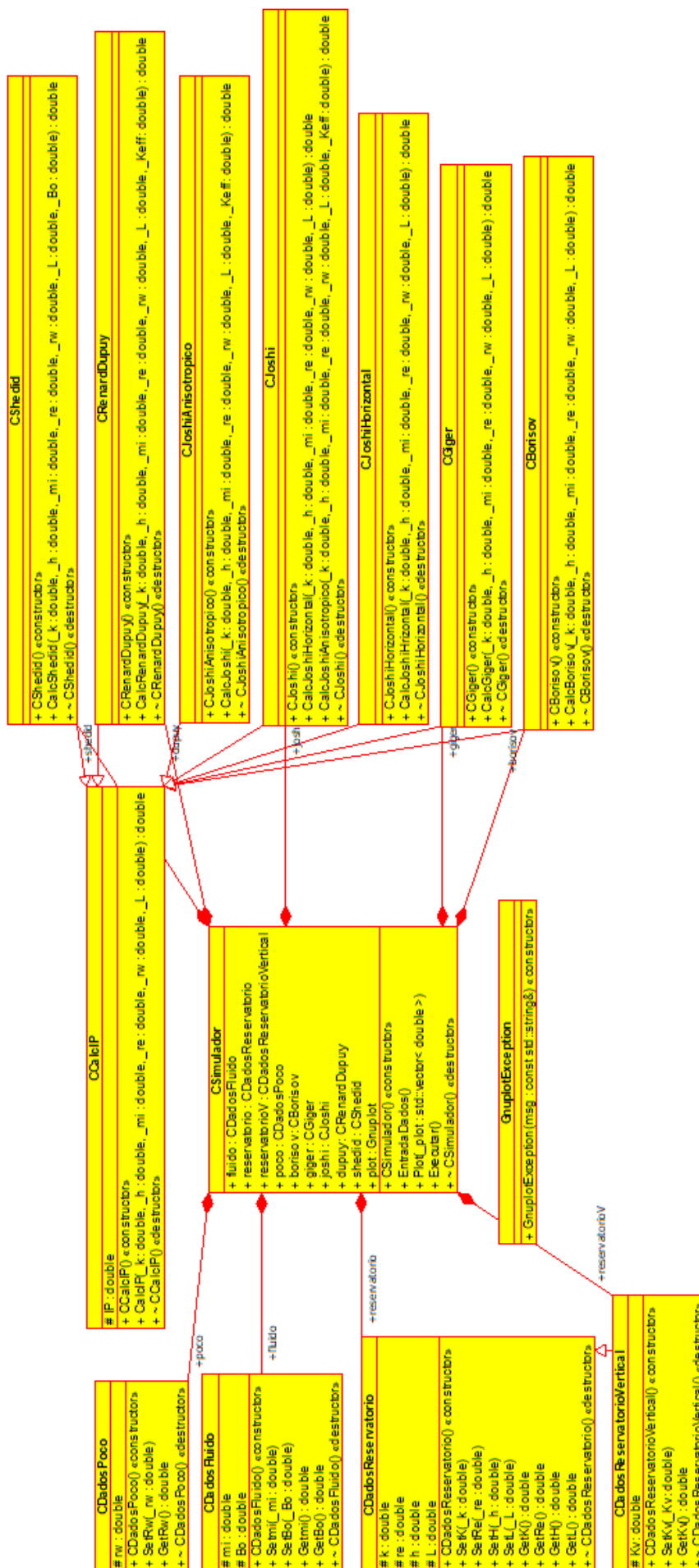


Figura 4.1: Diagrama de classes

4.1.1 Dicionário de classes

- Classe CDadosPoco: Solicita ao usuário os dados do poço e armazena.
- Classe CDadosFluido: Solicita ao usuário os dados do fluido e armazena.
- Classe CDadosReservatorio: Solicita ao usuário os dados do reservatório e armazena.
- Classe CDadosReservatorioVertical: Solicita ao usuário os dados do reservatório vertical e armazena.
- Classe CShedid: Calcula o IP do poço pelo modelo de Shedid.
- Classe CRenardDupuy: Calcula o IP do poço pelo modelo de Renard&Dupuy.
- Classe CJoshiAnisotropico: Calcula o IP do poço pelo modelo de Joshi em casos de reservatórios anisotrópicos.
- Classe CJoshi: Calcula o IP do poço pelo modelo de Joshi.
- Classe CJoshiHorizontal: Calcula o IP do poço horizontal pelo modelo de Joshi.
- Classe CGiger: Calcula o IP do poço pelo modelo de Giger.
- Classe CBorisov: Calcula o IP do poço pelo modelo de Borisov.
- Classe CCalcIP: Calcula o IP, utilizando os modelos disponíveis.
- Classe CSimulador: Faz as simulações do índice de produtividades dos poços (classe *main* do programa)
- Classe GnuplotExcrption: Gera uma visualização gráfica dos resultados usando software externo Gnuplot.

4.2 Diagrama de seqüência – eventos e mensagens

O diagrama de seqüência enfatiza a troca de eventos, de mensagens e sua ordem temporal. Contém informações sobre o fluxo de controle do software. Costuma ser montado a partir de um diagrama de caso de uso e estabelece o relacionamento dos atores (usuários e sistemas externos) com alguns objetos do sistema.

4.2.1 Diagrama de seqüência geral

O diagrama de seqüência geral do software é mostrado na figura 4.2.

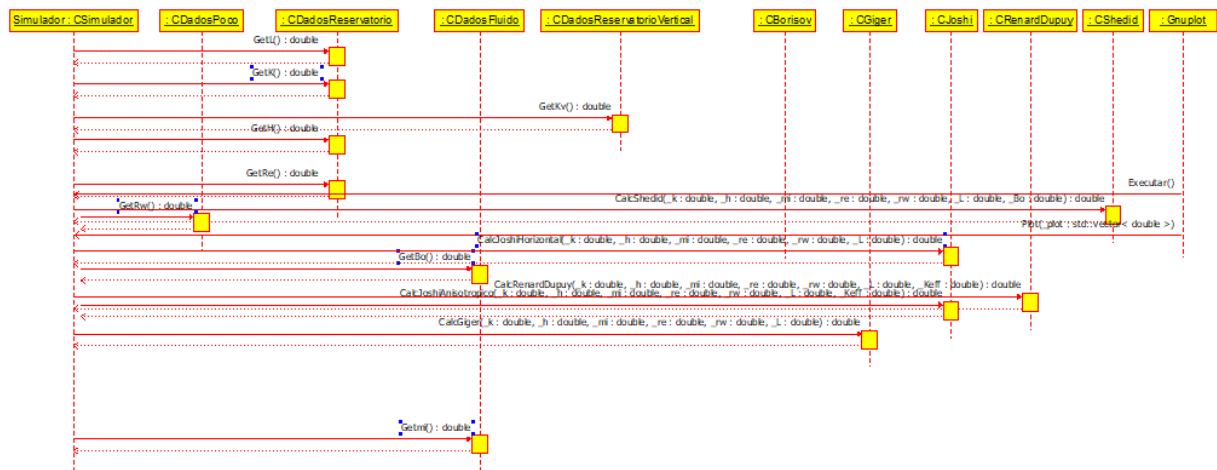


Figura 4.2: Diagrama de seqüência geral

4.3 Diagrama de comunicação – colaboração

No diagrama de comunicação o foco é a interação e a troca de mensagens e dados entre os objetos. Na figura 4.3 o diagrama de comunicação mostra a sequência do software para um caso em que o modelo de Joshi é utilizado para o cálculo do IP. Observe que a Classe CSimulador acessa os parâmetros do fluido, do poço e do reservatório a partir das classes CDadosFluido, CDadosPoco e CDadosReservatorio, respectivamente, que passa os atributos informados pelo usuário para as classes CCalcIP que por sua vez chama a classe CJoshi. A classe CSimulador então, após realizar os cálculos, envia esses resultados para a classe CGnuplotException que gera a visualização e salva esses resultados de forma gráfica.

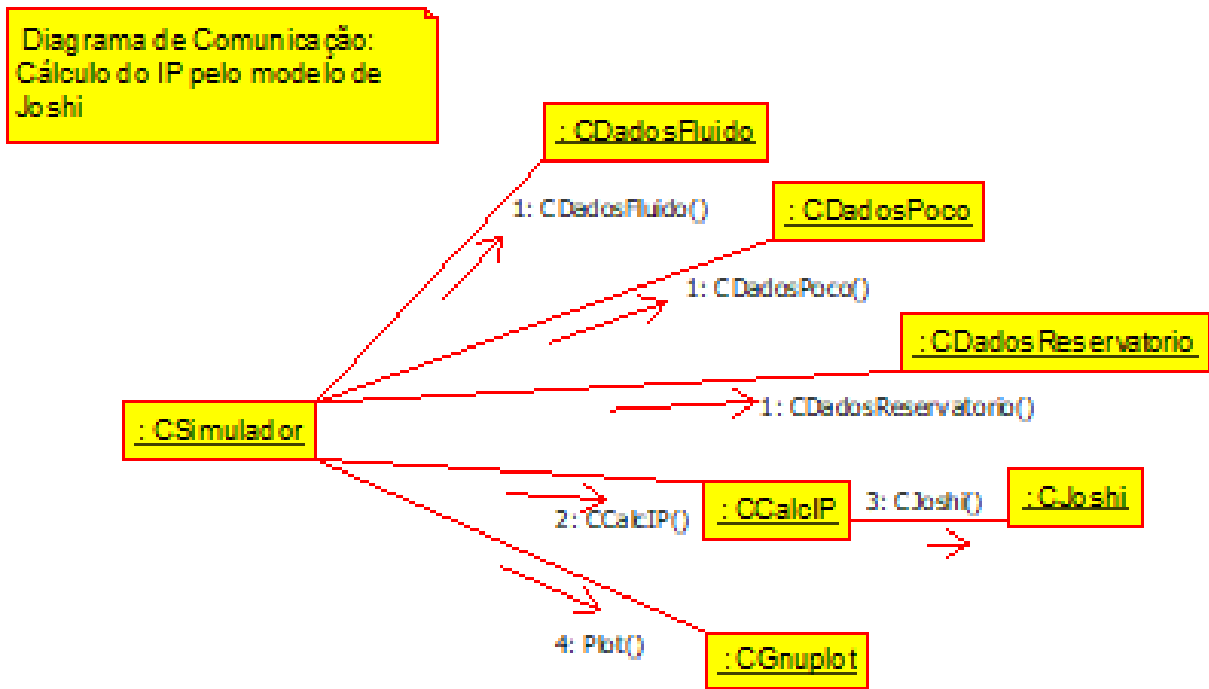


Figura 4.3: Diagrama de comunicação

4.4 Diagrama de máquina de estado

Um diagrama de máquina de estado representa os diversos estados que o objeto assume e os eventos que ocorrem ao longo de sua vida ou mesmo ao longo de um processo (histórico do objeto). É usado para modelar aspectos dinâmicos do objeto, como mostra a figura 4.4. Observe que, durante a execução do programa, o objeto passa por várias etapas.

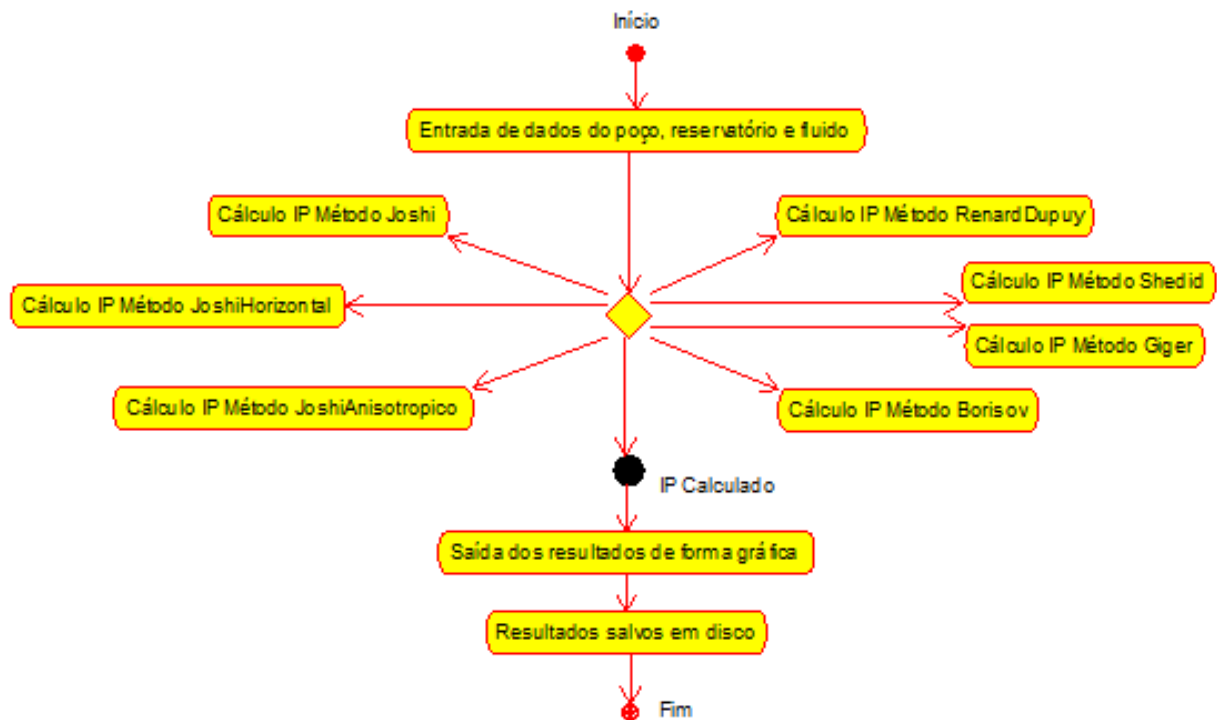


Figura 4.4: Diagrama de máquina de estado

4.5 Diagrama de atividades

O diagrama de atividades da figura 4.5 corresponde a uma atividade específica do diagrama de máquina de estado. Observe que foi escolhido um cenário fictício qualquer em que o poço recebe os dados do poço, raio interno, a viscosidade e fator volume formação do fluido e parâmetros do reservatório como raio externo, espessura, comprimento e permeabilidade vertical, que são informações necessárias para calcular IP. O modelo escolhido como exemplo é o Borisov.

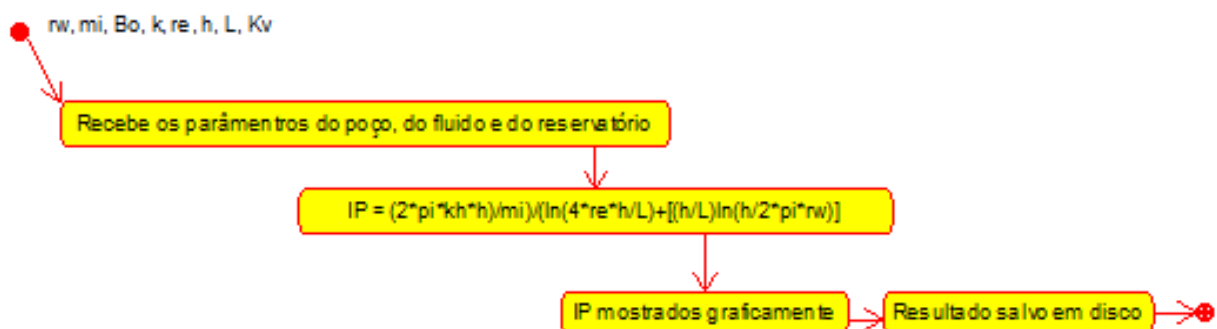


Figura 4.5: Diagrama de atividades

Capítulo 5

Projeto

Neste capítulo do projeto de engenharia veremos questões associadas ao projeto do sistema, incluindo protocolos, recursos, plataformas suportadas, implicações nos diagramas feitos anteriormente, diagramas de componentes e implantação. Na segunda parte revisamos os diagramas levando em conta as decisões do projeto do sistema.

5.1 Projeto do sistema

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

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

Segundo [Rumbaugh et al., 1994, Blaha and Rumbaugh, 2006], o projeto do sistema é a estratégia de alto nível para resolver o problema e elaborar uma solução. Você deve se preocupar com itens como:

1. Protocolos

- No software, o usuário poderá escolher se irá entrar com os dados do problema de forma manual ou se a entrada de dados será feita de modo que se importe arquivos no formato ASCII com extensão .txt.
- A interface utilizada será em modo texto.
- O software irá gerar saída de arquivos no formato ASCII com extensão .txt.

2. Recursos

- Neste projeto,o programa irá necessitar de utilizar os componentes internos do computador, como, por exemplo, HD, processador, mouse e teclado.
- Os gráficos serão gerados no programa externo Gnuplot.

3. Controle

- Neste projeto,o controle será sequencial.
- Não irá haver necessidade de otimização, pois o software e seus componentes trabalham com dados pequenos.
- Identificação e definição de *loops* de controle e das escalas de tempo.
 - Não se aplica.

4. Plataformas

- O software irá funcionar nos sistema operacionais Windows e GNU/Linux, sendo desenvolvido no Windows e testado no Windows e GNU/Linux.
- A linguagem de programação padrão utilizada é C++.
- As bibliotecas que serão utilizadas neste projeto são: iomanip, iostream, ostream, string, vector, entre outras.
- O projeto será totalmente desenvolvido na IDE Dev C++ na versão 5.11.

5. Padrões de projeto

- Normalmente,os padrões de projeto são identificados e passam a fazer parte de uma biblioteca de padrões da empresa.Entretanto, isso só ocorre após a realização de diversos projetos.Portanto, não se aplica neste caso.

5.2 Projeto orientado a objeto – POO

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

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

Efeitos do projeto no modelo estrutural

- Adicionar nos diagramas de pacotes as bibliotecas e subsistemas selecionados no projeto do sistema (exemplo: a biblioteca gráfica selecionada).
 - Neste projeto foi adicionada a biblioteca gráfica CGnuplot.
- Novas classes e associações oriundas das bibliotecas selecionadas e da linguagem escolhida devem ser acrescentadas ao modelo.
 - Não se aplica a este projeto.
- Estabelecer as dependências e restrições associadas à plataforma escolhida.
 - O *Software* necessita das plataformas GNU/Linux ou Windows para ser executado.
 - No sistema operacional Windows, é necessário a instalação do *Software* Gnuplot para o funcionamento do programa.

Efeitos do projeto no modelo dinâmico

- Revisar os diagramas de sequência e de comunicação considerando a plataforma escolhida.
 - Após necessidade de criação de uma classe CCalcIP que acessa todas as demais classes que calcula o índice de produtividade dos poços de acordo com modelos específicos, o diagrama de sequência precisou ser revisado e a sequência alterada para inclusão dessa etapa em que a CCalcIP acessasse os cálculos das classes herdeiras.
 - O mesmo se aplica ao diagrama de comunicação, com a inclusão dessa classe genérica CCalcIP a comunicação entre as classes do programa precisou ser alterada.
- Verificar a necessidade de se revisar, ampliar e adicionar novos diagramas de máquinas de estado e de atividades.
 - Houve necessidade de revisar, por motivos de mudanças decorridas na forma de construção do código que alterou a sequência de alguns eventos. A classe que antes calculava diretamente o índice de produtividade a partir do modelo específico escolhido pelo usuário agora faz parte das muitas classes herdeiras que são acessadas pela classe “mãe” CCalcIP que calcula todos o índice de produtividade a partir de todos os métodos, permitindo uma comparação entre eles e definição de qual o melhor design de poço em termos de produtividade.

Efeitos do projeto nos atributos

- Atributos novos podem ser adicionados a uma classe, como, por exemplo, atributos específicos de uma determinada linguagem de softwareção (acesso a disco, ponteiros, constantes e informações correlacionadas).
 - O atributo de acesso ao disco precisou ser incluído durante a elaboração do código para que o usuário pudesse inserir os dados do poço, do reservatório e do fluido (dados de entrada) em um arquivo .txt utilizando-o como input no programa. Esse atributo também está sendo utilizado ao final da execução, pela classe CGnuplot que além de gerar os gráficos comparando os difenretes métodos também o salva como imagem em disco.

Efeitos do projeto nos métodos

- Em função da plataforma escolhida, verifique as possíveis alterações nos métodos. O projeto do sistema costuma afetar os métodos de acesso aos diversos dispositivos (exemplo: hd, rede).
 - Não houve necessidade de alteração dos métodos.
- Algoritmos complexos podem ser subdivididos. Verifique quais métodos podem ser otimizados. Pense em utilizar algoritmos prontos como os da STL (algoritmos genéricos).
 - Não se aplica.
- Responda a pergunta: os métodos da classes estão dando resposta às responsabilidades da classe?
 - Os métodos que foram construídos estão gerando resultados coerentes com o que é abordado na literatura.
- Revise os diagramas de classes, de seqüência e de máquina de estado.
 - Foram realizadas várias revisões dos diagramas a medida que o código foi sendo construído e consequentemente havendo necessidade de tais alterações. O número de classes também mudou ao londo do processo, chegando à versão final que é a apresentadas neste documento.

Efeitos do projeto nas heranças

- Reorganização das classes e dos métodos (criar métodos genéricos com parâmetros que nem sempre são necessários e englobam métodos existentes).
 - Está sendo realizada uma reformulação das classes, separando-as em classes menores e conceitos independentes. Por exemplo, tínhamos elaborado uma classe para cada modelo de cálculo de IP e agora rearranjamos para que fique uma classe reunindo os modelos para poços do tipo horizontal e uma outra classe com os do tipo vertical.
 - Além disso, foi criada uma classe genérica de cálculo de IP para que ela seja acessada pela CSimulador e a partir daí acessar as classes herdeiras que calculam o IP a partir de modelos específicos. Anteriormente a CSimulador acessava diretamente todas essas classes.
- Abstração do comportamento comum (duas classes podem ter uma superclasse em comum).
 - Não se aplica a este projeto.
- Utilização de delegação para compartilhar a implementação (quando você cria uma herança irreal para reaproveitar código). Usar com cuidado.
 - Não se aplica a este projeto.
- Revise as heranças no diagrama de classes.
 - Foi criado relacionamento de herança entre a classe genérica CCalcIP e as demais CJosh, CGiger.. que calculam o IP a partir de um modelo específico.

Efeitos do projeto nas associações

- Deve-se definir na fase de projeto como as associações serão implementadas, se obedecerão um determinado padrão ou não.
 - As associações foram criadas e modificadas ao longo do desenvolvimento do código, respeitando a hierarquia das classes.
- Se existe uma relação de "muitos", pode-se implementar a associação com a utilização de um dicionário, que é um mapa das associações entre objetos. Assim, o objeto A acessa o dicionário fornecendo uma chave (um nome para o objeto que deseja acessar) e o dicionário retorna um valor (um ponteiro) para o objeto correto.
 - Não se aplica a este projeto.

- Evite percorrer várias associações para acessar dados de classes distantes. Pense em adicionar associações diretas.
 - Não se aplica a este projeto. Só houve criação de associações diretas.

Efeitos do projeto nas otimizações

- Faça uma análise de aspectos relativos à otimização do sistema. Lembrando que a otimização deve ser desenvolvida por analistas/desenvolvedores experientes.
 - Inicialmente pensamos em solicitar ao usuário os dados de entrada via terminal, ao longo do desenvolvimento implementamos a funcionalidade de colocar os dados em um arquivo externo que será lido pelo programa ao ser executado.
- Identifique pontos a serem otimizados em que podem ser utilizados processos concorrentes.
 - Não identificamos.
- Se o acesso a determinados objetos (atributos/métodos) requer um caminho longo (exemplo: A->B->C->D.atributo), pense em incluir associações extras (exemplo: A-D.atributo).
 - Não se aplica a este projeto, todos os atributos estão sendo acessados de forma direta.
- Revise as associações nos diagramas de classes.
 - Foram revisadas a medida que desenvolvemos o código.

5.3 Diagrama de componentes

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

Veja na Figura 5.1 um exemplo de diagrama de componentes. Observe que este inclui muitas dependências, ilustrando as relações entre os arquivos.

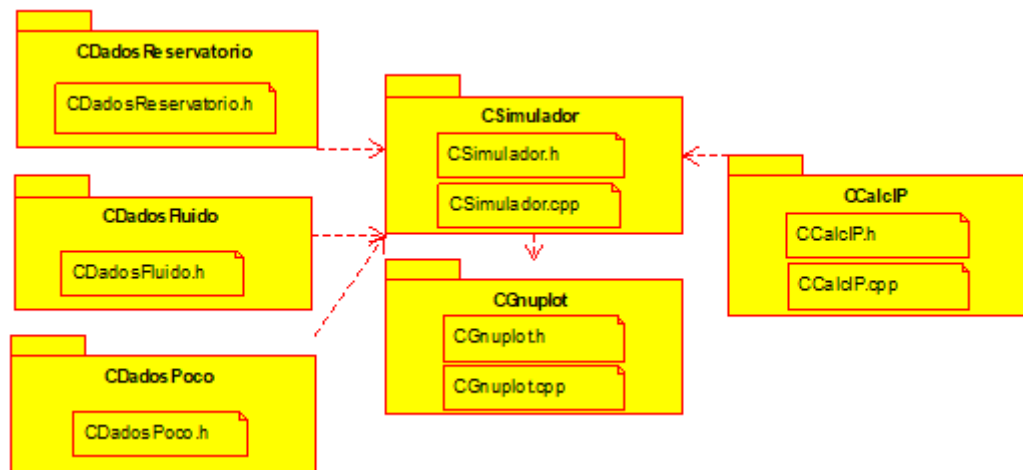


Figura 5.1: Diagrama de componentes

5.4 Diagrama de implantação

O diagrama de implantação é um diagrama de alto nível que inclui relações entre o sistema e o hardware e que se preocupa com os aspectos da arquitetura computacional escolhida. Seu enfoque é o hardware, a configuração dos nós em tempo de execução.

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

Veja na Figura 5.2 um exemplo de diagrama de implantação utilizado.

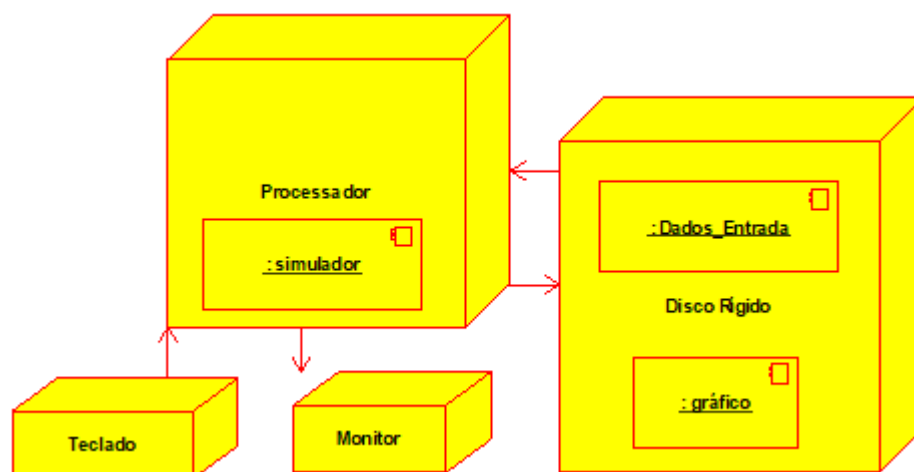


Figura 5.2: Diagrama de implantação

Capítulo 6

Implementação

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

Nota: os códigos devem ser documentados usando padrão **javadoc**. Posteriormente usar o programa **doxygen** para gerar a documentação no formato html.

- Veja informações gerais aqui <http://www.doxygen.org/>.
- Veja exemplo aqui <http://www.stack.nl/~dimitri/doxygen/manual/docblocks.html>.

Nota: ao longo deste capítulo usamos inclusão direta de arquivos externos usando o pacote *listings* do L^AT_EX. Maiores detalhes de como a saída pode ser gerada estão disponíveis nos links abaixo.

- http://en.wikibooks.org/wiki/LaTeX/Source_Code_Listings.
- <http://mirrors.ctan.org/macros/latex/contrib/listings/listings.pdf>.

6.1 Código fonte

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

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

Listing 6.1: Arquivo de cabeçalho da classe CAplicacao.

```
1// Este programa exemplifica a estrutura de um programa típico em C
  ++.
2// Note que no arquivo .h documentamos a interface; a forma de uso;
3// No arquivo .cpp detalhes dos códigos; lógica numérica-
  computacional.
4#include <string>
5#include <vector>
```

```
6
7/** Breve descrição da classe termina com ponto.
8 * ...descrição detalhada da classe...
9 * ...pode ter várias linhas...
10**/
11class CAplicacao
12{
13public:
14    /// Descrição breve do construtor.
15    /** Descrição detalhada do construtor.
16     * ....blablabla....
17     */
18    CAplicacao()        {};
19
20    /// Descrição breve do construtor.
21    /** Descrição detalhada do construtor.
22     * ....blablabla....
23     */
24    ~CAplicacao()       {};
25
26    /// Apenas exibe mensagem na tela.
27    void Run();
28
29    /// Seta valor de x
30    void X( int _x)      { x = _x; }
31
32    /// Retorna valor de x
33    int X()              { return x; }
34
35private:
36    /// Descrição breve do método M1.
37    /**
38     * Descrição detalhada....
39     * Posso incluir informações sobre parâmetros e retorno.
40     * @param a um inteiro que representa ....
41     * @param s uma string que representa ....
42     * @return retorna ...
43     */
44    int M1(int a, std::string s);
45
46    /// Descrição breve do atributo...
47    /** Descrição detalhada do atributo... */
```

```

48     std::vector<int> vy;
49
50     /// Descrição breve do atributo...
51     int x;
52
53     int z; ///< Descrição breve (use apenas se for bem curta!).
54
55     /// Enum representa (descrição breve).
56     /** Descrição detalhada. */
57     enum Enum {
58         EVal1, ///< Breve descrição EVal1.
59         EVal2, ///< Breve descrição EVal2.
60         EVal3  ///< Breve descrição EVal3.
61     } ;
62
63     /// Descrição breve.
64     /** Descrição detalhada. */
65     Enum    variavelDoTipoEnumeracao;
66
67 };

```

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

Listing 6.2: Arquivo de implementação da classe `CAplicacao`.

```

1// Este programa exemplifica a estrutura de um programa típico em
   C++
2#include <iostream>
3
4// Inclui a declaração da classe
5#include "CAplicacao.h"
6
7/** Note que no arquivo .cpp não é necessário colocar novamente a
   documentação
8 * que foi colocada no arquivo .h.
9 * A documentação no arquivo .cpp costuma usar o padrão básico de C
   ++ que é //
10 * e costuma estar mais diretamente relacionada a implementação em
   sí,
11 * ou seja, aos detalhes numéricos e computacionais;
12 * detalhes e explicação das contas e da lógica computacional.
13 * */
14void CAplicacao::Run()
15{

```

```
16 // std::cout escreve na tela o texto "Bem-vindo ao C++!"
17 std::cout << "Bem-vindo ao C++!" << std::endl;
18 }
```

Apresenta-se na listagem 6.3 o programa que usa a classe `CAplicacao`.

Listing 6.3: Arquivo de implementação da função `main()`.

```
1
2 /** Este programa exemplifica a estrutura/layout de um programa
   típico em C++ */
3
4 // Inclui o arquivo "CAplicacao.h" que tem a declaração da classe
   CAplicacao
5 #include "CAplicacao.h"
6
7 /// A função main(), retorna um inteiro, se chama main() e nao tem
   nenhum parametro
8 int main ()
9 {
10  CAplicacao ap; // Cria objeto do tipo CAplicacao com nome ap
11
12  ap.Run ();      // Executa o método Run() do objeto ap
13
14  return 0;       // A função main() deve retornar um inteiro
15                  // o zero indica que o programa terminou bem.
16 }
1 Bem vindo ao C++!
```

Nota:

Não perca de vista a visão do todo; do projeto de engenharia como um todo. Cada capítulo, cada seção, cada parágrafo deve se encaixar. Este é um diferencial fundamental do engenheiro em relação ao técnico, a capacidade de desenvolver projetos, de ver o todo e suas diferentes partes, de modelar processos/sistemas/produtos de engenharia.

Referências Bibliográficas

- [Blaha and Rumbaugh, 2006] Blaha, M. and Rumbaugh, J. (2006). *Modelagem e Projetos Baseados em Objetos com UML 2*. Campus, Rio de Janeiro. 19
- [JOSHI, 1988] JOSHI, S. D. (1988). *Production Forecasting Methods for Horizontal Wells*. SPE 17850, Tianjin, China. 8, 9
- [ROSA, 2006] ROSA, A. (2006). *Engenharia de Reservatórios de Petróleo*. Editora Interciencia, Rio de Janeiro. 9
- [Rumbaugh et al., 1994] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorenzen, W. (1994). *Modelagem e Projetos Baseados em Objetos*. Edit. Campus, Rio de Janeiro. 19
- [SHEDID, 2001] SHEDID, S. A. (2001). *Sensitivity Analysis of Horizontal Well Productivity under Steady-State Conditions*. SPE 72121, Kuala Lumpur. 10

Índice Remissivo

A

Análise orientada a objeto, 13
AOO, 13
Associações, 23
atributos, 22

C

Casos de uso, 5
colaboração, 16
comunicação, 16
Concepção, 3
Controle, 20

D

Diagrama de colaboração, 16
Diagrama de componentes, 24
Diagrama de execução, 25
Diagrama de máquina de estado, 17
Diagrama de sequência, 15

E

Efeitos do projeto nas associações, 23
Efeitos do projeto nas heranças, 23
Efeitos do projeto nos métodos, 22
Elaboração, 7
especificação, 3
Especificações, 3
estado, 17
Eventos, 15

H

Heranças, 23
heranças, 23

I

Implementação, 26

M

Mensagens, 15
métodos, 22
modelo, 21

O

otimizações, 24

P

Plataformas, 20
POO, 20
Projeto do sistema, 19
Projeto orientado a objeto, 20
Protocolos, 19

R

Recursos, 19