Capítulo 1

Projeto

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

1.1 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.

O projeto do sistema é a estratégia de alto nível para resolver o problema e elaborar uma solução. O presente projeto foi desenvolvido para apresentar soluções gráficas e via arquivo .txt.

1. Protocolos

    - Definição dos protocolos de comunicação entre os diversos elementos externos
        - Neste projeto o software irá se comunicar com o componente externo GNUPLOT, que tem como finalidade plotar um gráfico da queda de pressão em função do tempo para analisar o comportamento da permeabilidade a medida que ocorre a injeção de água de baixa salinidade.
        - A entrada de dados referente as propriedades da rocha, da partícula e do fluído será efetuada no formato ascii, utilizando a extensão .txt. Enquanto

que outros dados a entrada será através do teclado.

- Definição do formato dos arquivos gerados pelo software.
  - Neste projeto será criado uma pasta com os resultados ao final de sua execução. Nessa pasta será gerado vários arquivos .txt contento as concentrações de acordo com o espaço(x) e o tempo, cada arquivo .txt é referente a um tempo específico, variando apenas o x.
  - O programa salva as imagens dos gráficos no disco no formato .png.

2. Recursos

- Identificação e alocação dos recursos globais, como os recursos do sistema serão alocados, utilizados, compartilhados e liberados. Implicam modificações no diagrama de componentes.
  - Neste projeto será utilizado o HD, o processador, o teclado, a memória, a tela e os demais componentes internos do computador.
  - Neste pojeto será utilizado um banco de dados no forrmato .txt.
- Identificação da necessidade do uso de banco de dados. Implicam em modificações nos diagramas de atividades e de componentes.
  - Neste projeto será necessário o uso do GNUPLOT para plotar o gráfico.

3. Controle

- Identificação da necessidade de otimização. Por exemplo: prefira sistemas com grande capacidade de memória; prefira vários hds pequenos a um grande.
  - Não há necessidade otimização. Porém, dependendo da quantidade do número de pontos o programa pode ser otimizado.
- Identificação e definição de *loops* de controle e das escalas de tempo.
  - O tempo e o espaço é dividido de forma igualmente espaçada.

4. Plataformas

- Identificação e definição das plataformas a serem suportadas: hardware, sistema operacional e linguagem de software.
  - A linguagem utilizada neste projeto será C++, desse modo o programa será multiplataforma. Pode ser executado no Mac OS X, GNU/Linux e Windows.
- Seleção das bibliotecas externas a serem utilizadas.

  - Neste projeto será utilizada a biblioteca padrão da linguagem de C++, incluindo vector, iostream, string, math.h, cstddef, fstream, iomanip. A partir da classe CGNUPLOT contendo a biblioteca externa CGNUPLOT será permitido ter acesso ao programa GNUPLOT.

- Seleção do ambiente de desenvolvimento para montar a interface de desenvolvimento – IDE.
  - O software DEV C++ será o ambiente de desenvolvimento utilizado para montar a interface de desenvolvimento do programa. O programa será utilizado no sistema operacional do Windows 10 de 64 bits. O compilador será o gcc/g++.

1.2 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.

Por exemplo: na análise você define que existe um método para salvar um arquivo em disco, define um atributo nomeDoArquivo, mas não se preocupa com detalhes específicos da linguagem. Já no projeto, você inclui as bibliotecas necessárias para acesso ao disco, cria um objeto específico para acessar o disco, podendo, portanto, acrescentar novas classes àquelas desenvolvidas na análise.

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 a biblioteca gráfica selecionada foi a CGnuplot, sendo necessária a instalação do software Gnuplot para plotar os gráficos.
  - vector: armazenamento de dados por meio de vetores.
  - iostream: entrada e saida de dados, pelo teclado e pela tela respectivamente.
  - string: utilizada para conversão de um valor número em uma sequência de caracteres.

  - cmath.h: possui funções matemáticas visando a realização de cálculos.
  - fstream: estabelece um canal de comunicação entre o arquivo e o programa. Utilizado para gravar e ler arquivos de disco.
  - iomanip: fornece recursos para manipular a formatação de saída.

- Novas classes e associações oriundas das bibliotecas selecionadas e da linguagem escolhida devem ser acrescentadas ao modelo.

  - Neste projeto o usuário acessa o arquivo dos dados de entrada por meio do nome dos arquivos .txt, utilizando as bibliotecas fstream, string etc.

- Estabelecer as dependências e restrições associadas à plataforma escolhida.

  - O software pode ser executado por meio das plataformas: GNU/Linux, MAC OS ou Windows.

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).

  - Neste projeto foi necessário adicionar alguns atributos e inserir via teclado para facilitar a alteração e execução do programa.

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).

  - Neste projeto foi necessário adicionar um método para salvar em disco o resultado das integrais para teste dos valores obtidos com os valores reais, a fim de validar o método de númerico de integração.
  - Métodos para salvar em arquivo .dat foram adicionados, a fim de obter todos os valores calculados e obtidos permitindo comparação e cálculo para veirificação.

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).

  - Foi realizado uma reorganização das classes para facilitar o acesso ao banco de dados, dessa forma foi criada uma herança para que a Classe CSimuladorParticulas consiga herdar todos os parâmetros da Classe CRocha e CParticulaFluido, visando acelerar o acesso às variáveis.

- Revise as heranças no diagrama de classes.

  - A classe CSimuladorParticulas é herdeira das classes CRocha e CParticulaFluido, a fim de acessar os valores das variáveis para cálculo das concentrações.
  - A classe MetodoSimpson é herdeira da classe MetodoIntegracaoNumerica1D. E as classes Funcao_Sigma_n_diferente_1, Funcao_Sigma_n_1 são herdeiras da classe Funcao1x1.

Efeitos do projeto nas associações

- Não houve alteração nas associações do projeto.

Efeitos do projeto nas otimizações

- Não foi necessário rever nesta esta do projeto.

1.3 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 1.1 um exemplo de diagrama de componentes. Observe que este inclui muitas dependências, ilustrando as relações entre os arquivos. Por exemplo: o subsistema biblioteca inclui os arquivos das classes A e B, e a geração dos objetos A.obj e B.obj depende dos arquivos A.h, A.cpp, B.h e B.cpp. A geração da biblioteca depende dos arquivos A.obj e B.obj. O subsistema biblioteca Qt, um subsistema exerno, inclui os arquivos de código da biblioteca Qt e a biblioteca em si. O subsistema banco de dados representa o banco de dados utilizado pelo sistema e tem uma interface de acesso que é utilizada pelo software para acesso aos dados armazenados no banco de dados. O software executável a ser gerado depende da biblioteca gerada, dos arquivos da biblioteca Qt, do módulo de arquivos MinhaJanela e do banco de dados.

Algumas observações úteis para o diagrama de componentes:

- De posse do diagrama de componentes, temos a lista de todos os arquivos necessários para compilar e rodar o software.

- Observe que um assunto/pacote pode se transformar em uma biblioteca e será incluído no diagrama de componentes.
- A ligação entre componentes pode incluir um estereótipo indicando o tipo de relacionamento ou algum protocolo utilizado.

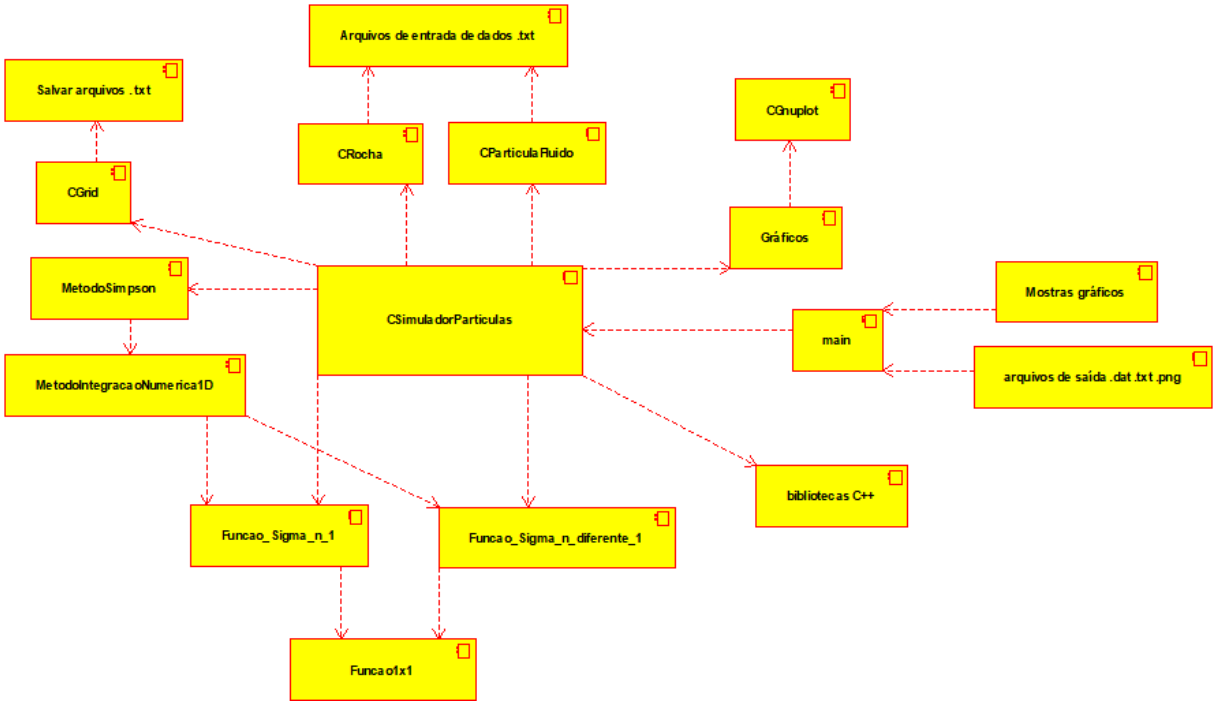

Figura 1.1: Diagrama de componentes

1.4 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.

A Figura 1.2 mostra o diagrama de implantação utilizado. Os dados foram obtidos da literatura e foram armazenados em arquivos .txt no computador. O programa importa os arquivos para acessar os dados e utiliza o monitor para comunicar com o usuário. Os gráficos e arquivos obtidos são salvos no disco rígido.

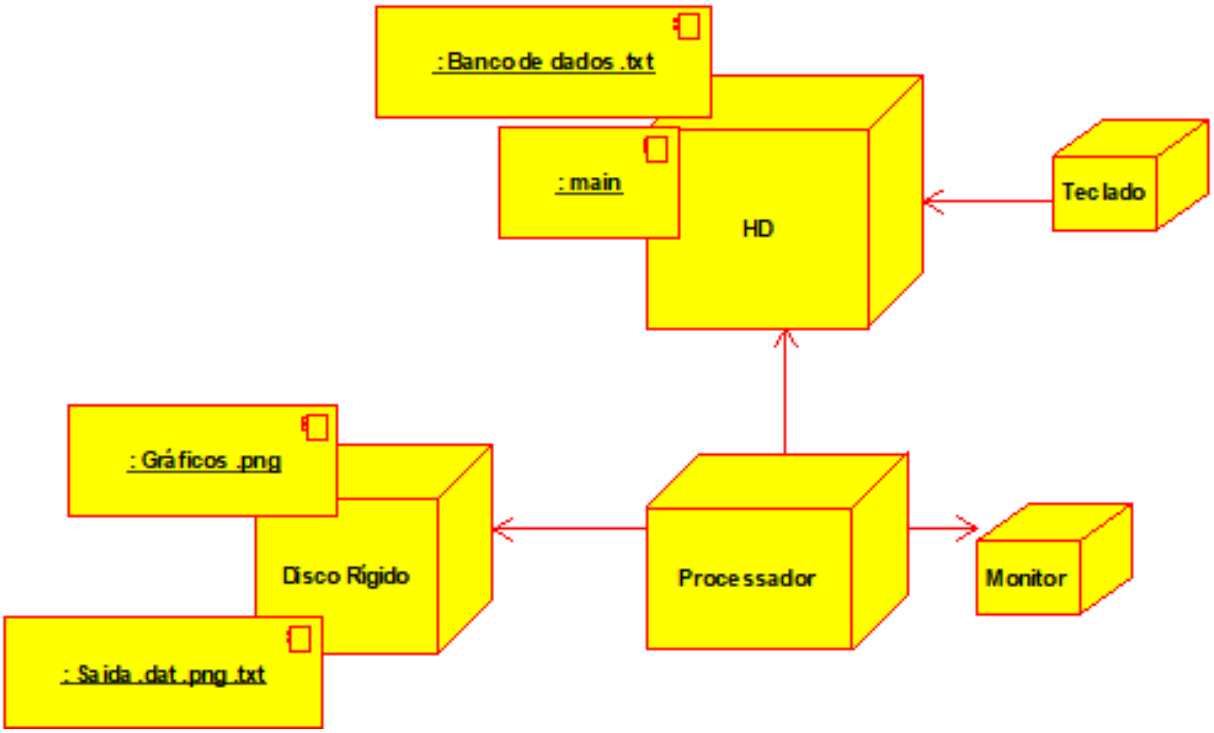

Figura 1.2: Diagrama de implantação