Projeto de Software Orientado a Objetos

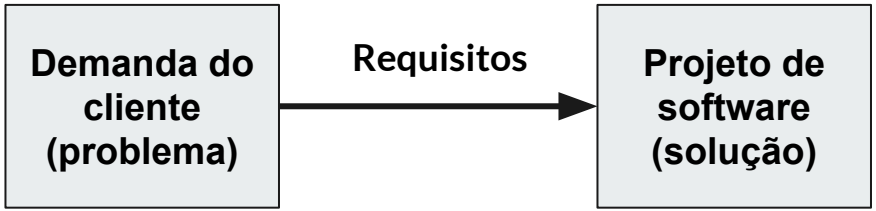
Projeto de Software

Fabio Morais

Projeto de Software

Etapa central do processo de desenvolvimento de software

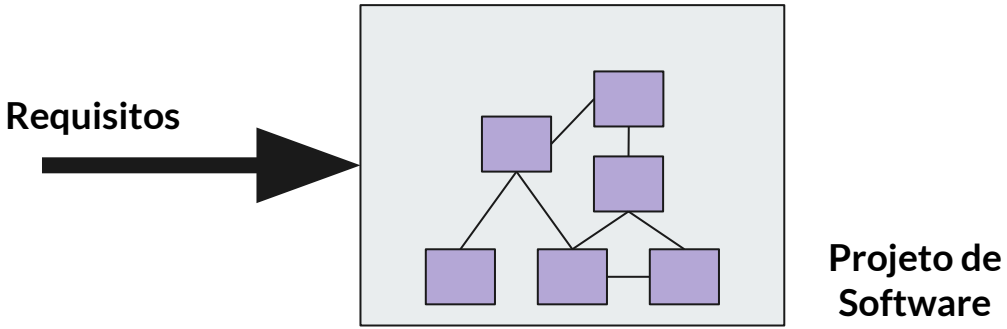
- **Análise**: requisitos do cliente (problema) → **Especificação**
- **Projeto**: como será implementado (solução) → **Software**



Projeto de Software

Identificar **componentes** de software e seus **relacionamentos**

- Com base nos requisitos especificados (Funcionais e Não Funcionais)



Construção do Projeto

Definição e detalhamento de aspectos importantes:

- Contexto e interações externas
- Arquitetura do sistema
- Principais classes / objetos
- Modelagem dos componentes
- Interfaces de acesso

Projeto de Software

Redução da **complexidade** do sistema de software

- **Requer experiência e criatividade no processo**

Considerar **propriedades**, **princípios e padrões** de projeto de software ajuda no processo

Propriedades

- Integridade conceitual
  - Coerência de funcionalidades, nomenclatura, formato, etc
- Encapsulamento (ocultamento da informação)
  - Operações definidas para a manipulação de dados
- Coesão e Acoplamento

Padrões de Projeto de Software

- Problemas no desenvolvimento de software se repetem
- Boas soluções ... por bons desenvolvedores
- Reutilização das boas soluções em outros projetos
    - Padrões de projeto de software
    - Adaptados ao contexto do seu problema

Padrões de Projeto de Software

- Reutilização de **ideias** e **não de código**
  - Descrição de um problema recorrente
  - Base de solução para o problema descrito

Elementos essenciais de um padrão

- Nome
- Problema abordado
- Solução proposta
- Consequências

Nome

- Sugestivo / representativo do problema ou solução
- Deve ser possível identificar a finalidade pelo nome
- Exemplo: **Singleton**
    - Padrão utilizado para que se tenha **apenas uma instância** de uma classe

Nome

- Permite um nível mais elevado de abstração
    - Comunicação entre a equipe de desenvolvimento
    - Resume a documentação de código
    - Facilita a documentação arquitetural do sistema
    - Padronização dos nomes em inglês

```
/** Nesta classe utiliza-se um construtor
 * privado, com um método estático que
 * retorna a única instância desta
classe,
 * sincronizado para evitar que outra
 * instância seja recuperada por outra
 * linha de execução...
 * @author Nazareno
 * @version 1.0
 */
public class Calendar{
   …
```

```
/** Nesta classe implementa-se o
 * padrão Singleton...
 * @author Nazareno
 * @version 1.0
 */
public class Calendar{
   …
```

Problema abordado

- Quando aplicar um padrão?
- Em que situação o padrão melhora o projeto de software?
- Quando **não** se deve utilizar o padrão?

Problema e um contexto

- Identificação de um problema a ser resolvido
    - *Como implementar um sistema com interesse e notificação?*
- Verificar se o problema identificado é similar ao descrito
    - Aplicar o padrão para a solução do problema

Especificação de exemplos de aplicação

- Como definir uma instância única de uma classe?
- Como representar algoritmos como objetos?
- Como implementar hierarquias parte-todo?

Solução

- Elementos do projeto da solução, responsabilidades e colaborações
  - Modelo conceitual
  - Diagrama de classes
  - Diagrama de interação

Solução

- Não descreve uma implementação concreta
  - Modelo genérico de solução que pode ser reutilizado
  - Deverá ser adaptado ao contexto do problema
  - Apresenta um exemplo de solução a ser seguido

Consequências

- Resultados do uso do padrão
- Vantagens e desvantagens (visão crítica)
- Análise de impactos de uso do padrão
    - Flexibilidade, Extensibilidade, Reuso, Desempenho, ...

Atribuição de Responsabilidades

- Como atribuir responsabilidades?
  - Mantendo o sistema flexível e fácil de manter?
- **G**eneral **R**esponsibility **A**ssignment **S**oftware **P**atterns
  - Padrões **GRASP**
  - Dividir responsabilidade entre objetos

Atribuição de Responsabilidades - GRASP

- Responsabilidade de um objeto fazer algo
  - Ao próprio objeto ou a outros objetos
- Responsabilidade de conhecer/guardar algo
  - Dados encapsulados
  - Objetos relacionados

Responsabilidade ≠ Método

- Métodos implementam responsabilidades
  - Sabem fazer ou manipulam algo (dados ou objetos)
- Objetos colaboram para cumprir responsabilidades
  - Relações entre objetos (composição, agregação, etc)

GRASP são padrões e/ou princípios

- **Information Expert**
  - Quem é o especialista da informação?
- **Creator**
  - Quem deve criar uma determinada instância de classe?

GRASP são padrões e/ou princípios

- **Baixo Acoplamento (Low Coupling)**
  - Como diminuir o acoplamento entre classes?
- **Alta Coesão (High Cohesion)**
  - Como aumentar a coesão no sistema?
- Controller, Polymorphism, Pure Fabrication, Indirection e Protected Variations

Introdução a Projeto de SW

Projeto de Software - Fabio Morais