Projetos de Sistemas de Informação

Educação a Distância
Cruzeiro do Sul Educacional
Campus Virtual

Material Teórico

Padrões do Processo de Desenvolvimento de Software

Responsável pelo Conteúdo:
Prof. Ms. Fábio Peppe Beraldo.

Revisão Textual:
Prof. Ms. Claudio Brites.

UNIDADE Padrões do Processo de Desenvolvimento de Software

- Introdução
- A Estrutura dos Padrões
- Padrões de Criação
- Documentação

Objetivo de APRENDIZADO

- Esta unidade apresenta uma introdução às funções dos padrões, de cada um dos principais padrões será feita uma apresentação de seus métodos de trabalho.
- Para total aproveitamento, leia atentamente o conteúdo teórico e participe de todas as atividades – sem perder o prazo de fechamento da unidade. Tudo isso contribuirá para que você tenha perfeita compreensão do conteúdo disponibilizado.

Este guia de estudo demonstra a forma de organizar suas atividades para cumprir os propósitos da disciplina online objetiva salientar algumas informações importantes sobre esta Unidade, bem como sobre o estudo online em si.

Assim, é necessário organizar seu tempo para ler atentamente o material teórico e assistir os vídeos, se houver, inclusive do material complementar.

Especificamente quanto à Unidade III, o objetivo é:

- Apresentar o conceito de Padrões com foco em Desenvolvimento de Software;
- Conceituar a Estrutura dos Padrões;
- Conceituar os Padrões de Domínio Específicos;
- Classificar os Padrões;
- Padrões de Criação;
- Padrões Estruturais;
- Padrões de Comportamento;
- Padrões de Concorrência.
- Apresentar os conceitos de Documentação dos Padrões;
- Críticas à ideia de Padroes.

Contextualização

Em engenharia de software, um padrão de projeto é uma solução reutilizável para um problema no design de software. Um padrão de projeto não é um projeto acabado, que pode ser transformado diretamente em fonte ou código de máquina, é uma descrição ou modelo de como resolver um problema que pode ser utilizado em diversas situações diferentes.

Nesta unidade de Projeto de Sistemas de Informação, teremos uma visão de como os padrões desenvolvem as ações que o programador deve implementar na aplicação.

Introdução

Em engenharia de software, um padrão de projeto é uma solução reutilizável para um problema no design de software. Um padrão de projeto não é um projeto acabado que pode ser transformado diretamente em fonte ou código de máquina, é uma descrição ou modelo de como resolver um problema que pode ser utilizado em diversas situações diferentes. Os padrões desenvolvem as ações que o programador deve implementar na aplicação. Os padrões de projeto orientado a objetos tipicamente mostram relações e interações entre as classes ou objetos, sem especificar as classes de aplicações finais ou objetos que estão envolvidos. Padrões que implicam orientação a objetos não são tão aplicáveis em linguagens de programação funcional.

Existem muitos tipos de padrões de design:

- Padrões de estratégia: esse algoritmo abordar as preocupações relacionadas com as estratégias de alto nível, que descrevem como explorar características de aplicação em uma plataforma de computação;
- Padrões de projeto computacional: esses padrões abordam as preocupações relacionadas com a identificação das chaves de computação;
- Padrões de execução: tratam de preocupações relacionadas com a execução de aplicativos de apoio, incluindo estratégias de execução de fluxos de tarefas e blocos de construção para suportar a sincronização de tarefas;
- Padrões de implementação de estratégia: esses padrões abordam as preocupações relacionadas com a implementação de código fonte para apoio da(s):
  - » Organização do programa;
  - » Estruturas de dados comuns específicas para programação paralela.
- Padrões de projeto estruturais: abordam as preocupações relacionadas com estruturas de alto nível dos aplicativos que estão sendo desenvolvidos.

A Estrutura dos Padrões

Os padrões de design podem acelerar o processo de desenvolvimento, fornecendo testes para alguns paradigmas de desenvolvimento. O projeto eficaz de software deve considerar as questões que não são visíveis até fases futuras da implementação. Reutilizar os padrões de design ajuda a prevenir problemas sutis, que podem causar grandes problemas, e também a melhorar a legibilidade do código para programadores e arquitetos que estão familiarizados com os padrões.

Com o objetivo de alcançar maior flexibilidade, padrões de projeto geralmente introduzem níveis adicionais de erro, que em alguns casos podem complicar os projetos resultantes e prejudicar o desempenho do aplicativo.

Comumente, um padrão deve ser reprogramado para cada aplicativo que usá-lo – uma vez que alguns autores veem isso como um retrocesso na reutilização de software, tal como previsto por componentes, os pesquisadores trabalham para transformar padrões em componentes.

Técnicas de design de software são difíceis de aplicar a uma gama ampla de problemas. Os padrões de design fornecem soluções gerais, documentadas em um formato que não requer especificidades ligadas a um problema particular.

Padrões de projeto são compostos de várias seções, dentre as mais importantes estão: a estrutura, os participantes, e as seções de colaboração. Essas seções descrevem o design: a microarquitetura que os desenvolvedores copiam e adaptam a seus projetos particulares para resolver o problema recorrente descrito pelo padrão de design. A microarquitetura é um conjunto de componentes do programa (por exemplo: classes, métodos, etc.) e seus relacionamentos. Os desenvolvedores usam o padrão de design introduzindo em seus projetos essa microarquitetura protótipo, o que significa que as microarquiteturas em seus projetos terão estrutura e organização semelhantes ao motivo de design escolhido.

Padrões de Domínio Específicos

Muitos esforços têm sido feitos para codificar padrões de projeto em situações específicas, incluindo o uso de padrões de projeto existentes, bem como padrões de projeto específicos do domínio. Os exemplos incluem: os padrões de design de interface do usuário, a visualização da informação, design seguro, "usabilidade seguro", web design e design de modelo de negócio. A Conferência Anual de Linguagem de Programação determina diversos padrões de domínio específicos.

Classificação e Lista

Os padrões de projeto foram originalmente agrupados nas categorias: padrões de criação, padrões estruturais e padrões de comportamento, descritos como utilizando os conceitos de delegação, agregação e de consulta, para posteriormente serem usados em projetos orientados a objetos. Outra classificação também introduz a noção de padrão de arquitetura de design, que pode ser aplicada no nível de arquitetura do software.

Padrões de Criação

Os padrões de criação ou padrões de projeto criacionais são padrões de projeto que tratam de mecanismos de criação de objeto, tentando criar objetos de uma forma adequada à situação. A forma básica de criação do objeto pode resultar em problemas de projeto ou de complexidade adicionada ao projeto.

Os padrões de criação são compostos de duas ideias dominantes: uma delas é encapsular o conhecimento sobre quais classes concretas o sistema trabalha; a outra está em acoplar como instância dessas classes concretas criadas e combinadas.

Modelos do padrão	Descrição
Abstract Factory	Fornece uma interface para criar famílias de objetos relacionados ou dependentes sem especificar suas classes concretas.
Builder	Separa-se a construção de um objeto complexo da sua representação, permitindo ao mesmo processo de construção criar várias representações.
Factory Method	Define uma interface para criar um único objeto, mas deixa as subclasses decidirem qual classe instanciar. O modelo permite a instanciação de classe passar para subclasses.
Lazy Initialization	Tática de atrasar a criação de um objeto, o cálculo de um valor, ou algum outro processo caro – até a primeira vez que é necessário. Este padrão aparece no catálogo GoF como "procuração virtual", uma estratégia de implementação para o padrão Proxy.
Multiton	Certifique-se de que uma classe nomeou apenas exemplos, e forneça o ponto global de acesso a eles.
Object Pool	Evita aquisição cara e liberação de recursos através da reciclagem de objetos que não estão mais em uso. Pode ser considerada uma generalização do pool de conexões e padrões de pool of threads.
Prototype	Especifica os tipos de objetos para criar usando uma instância protótipo, e cria novos objetos copiando esse protótipo.
Resource Acquisition Is Initialization	Assegura que os recursos são devidamente liberados, amarrando-os para a vida útil de objetos adequados.
Singleton	Certifique-se de que uma classe tem apenas uma instância, e forneça um ponto global de acesso a ela.

Padrões Estruturais

Padrões de projeto estrutural são os padrões de design que facilitam o projeto através da identificação de uma maneira simples de perceber relações entre entidades.

Modelos do padrão	Descrição
Adapter ou Wrapper ou Translator	Converte a interface de uma classe em outra interface esperada pelos clientes. Um adaptador permite às classes trabalharem juntas, o que não seria possível por conta de interfaces incompatíveis. O padrão de integração empresarial equivalente é o tradutor.
Bridge	Desacoplar uma abstração de sua implementação permitindo aos dois variarem independentemente.
Composite	Compor objetos em estruturas de árvore para representar hierarquias parte-todo. O padrão composite permite que clientes tratem objetos individuais e composições de objetos uniformemente.
Decorator	Anexa responsabilidades adicionais a um objeto dinamicamente mantendo a mesma interface. Decoradores fornecem uma alternativa flexível para subclasses para extensão da funcionalidade.
Facade	Fornece uma interface unificada para um conjunto de interfaces em um subsistema. Facade define uma interface de nível mais elevado, que faz com que o subsistema seja mais fácil de usar.
Flyweight	Usar compartilhamento para suportar um grande número de objetos semelhantes de forma eficiente.
Front Controller	O padrão refere-se à concepção de aplicações web. Ele fornece um ponto de entrada centralizada para o tratamento dos pedidos.
Module	Grupo de vários elementos relacionados, tais como: aulas, métodos utilizados em nível mundial; em uma única entidade conceitual.
Proxy	Fornece um substituto ou espaço reservado para outro objeto, para controlar o acesso a ele.
Twin	Twin permite modelar heranças múltiplas em linguagens de programação que não suportam esse recurso.

Padrões de Comportamento

Padrões de projeto comportamentais são padrões de projeto que identificam padrões comuns de comunicação entre objetos e realizam esses padrões. Ao fazerem isso, esses padrões aumentam a flexibilidade na realização dessa comunicação.

Modelos do padrão	Descrição
Blackboard	Observador generalizado, que permite vários leitores e escritores. Comunica-se com todo o sistema de informação.
Chain Of Responsibility	Evita o acoplamento do remetente de uma solicitação ao seu receptor, dando a mais de um objeto a chance de lidar com o pedido. Coloca em cadeia os objetos que recebem e passam a solicitação ao longo dessa cadeia, até que um objeto o manipule.
Command	Encapsula uma solicitação como um objeto, permitindo assim parametrizar clientes com pedidos diferentes, fila ou solicitações de registro, e apoiar as operações que podem ser desfeitas.
Interpreter	Dada uma linguagem, definir uma representação para sua gramática juntamente com um interpretador, que usa a representação para interpretar sentenças na língua.
Iterator	Fornece uma maneira de acessar os elementos de um objeto agregado sequencialmente sem expor sua representação subjacente.
Mediator	Define um objeto que encapsula como um conjunto de objetos interagem. O modelo mediator promove o acoplamento fraco, permitindo objetos a se referirem uns aos outros de forma explícita, e que lhes permite variar a sua interação de forma independente.
Memento	Sem violar o encapsulamento, captura e externaliza o estado interno de um objeto, permitindo ao objeto ser restaurado para esse estado mais tarde.
Null Object	Evita referências nulas, fornecendo um objeto padrão.
Observer Or Publish/Subscribe	Define uma dependência um-para-muitos entre objetos, na qual uma mudança de estado em um objeto resulta na notificação e atualização de todos os seus dependentes automaticamente.
Servant	Define funcionalidade comum para um grupo de aulas.
Specification	Lógica de negócios recombinável de forma booleana.
State	Permite que um objeto altere seu comportamento quando seu estado interno muda. O objeto aparecerá para mudar sua classe.
Strategy	Define uma família de algoritmos, encapsula cada um, e os torna intercambiáveis. Essa estratégia permite que o algoritmo varie independentemente dos clientes que o utilizam.
Template Method	Definir o esqueleto de um algoritmo em uma operação, adiando alguns passos para subclasses. O modelo Template Method permite que subclasses redefinam determinadas etapas de um algoritmo sem alterar a estrutura do algoritmo.
Visitor	Representar uma operação a ser realizada sobre os elementos de uma estrutura de objeto. Visitor permite que você defina uma nova operação sem mudar as classes dos elementos sobre os quais opera.

Padrões de Concorrência

Padrões de concorrência são os tipos de padrões de projeto que lidam com o paradigma de programação multithreaded.

Um thread de execução é a menor sequência de instruções programadas que podem ser gerenciadas de forma independente por um programador do sistema operacional. O programador em si é um processo leve. A implementação de threads e processos difere de um sistema operacional para outro, mas, na maioria dos casos, um segmento está contido dentro de um processo. Vários segmentos podem existir dentro do mesmo processo e compartilhar recursos – como memória –, enquanto diferentes processos não compartilham desses recursos. Em particular, as threads de um processo compartilham as instruções desse (seu código) e seu contexto (os valores que suas variáveis fazem referência a qualquer momento).

O multithreading é geralmente implementado por "time-division multiplexing". Esta troca de contexto acontece com frequência em um multiprocessador ou sistema multicore. Segmentos podem ser verdadeiramente concorrente, com cada processador ou núcleo a executar uma thread separada simultaneamente.

Modelos do padrão	Descrição
Active Object	Desacopla a execução do método de invocação de método que reside em sua própria thread de controle. O objetivo é introduzir concorrência, usando o método assíncrono de invocação e um organizador para o tratamento dos pedidos.
Balking	Somente executa uma ação em um objeto quando o objeto está em um estado particular.
Binding Properties	Combina vários observadores para forçar propriedades em diferentes objetos a serem sincronizados ou coordenados de alguma forma.
Double-Checked Locking	Reduz a sobrecarga de adquirir um bloqueio por primeiro teste do critério de bloqueio (a "dica de bloqueio") de forma insegura; somente se suceder do bloqueio real prosseguir. Pode ser perigoso quando implementado em algumas combinações de linguagem de hardware e, por conseguinte, ser considerado um antipadrão.
Event-Based Asynchronous	Aborda problemas com o padrão assíncrono que ocorrem em programas multithread.
Guarded Suspension	Gerencia operações que exigem tanto um bloqueio a ser adquirido quanto um pré-requisito a ser atendido antes que a operação possa ser executada.
Join	Associa padrões que auxiliam na forma de escrever programas concorrentes, paralelos e distribuídos pela mensagem que passa. Em comparação com a utilização de threads e locks, esse é um modelo de programação de alto nível.
Lock	Coloca um bloqueio (lock) em um recurso, impedindo a outros segmentos de acessá-lo ou modificá-lo.
Messaging Design Pattern (MDP)	Permite a troca de informações (ou mensagens) entre componentes e aplicações.
Monitor Object	Um objeto cujo os métodos estão sujeitos à exclusão recíproca, evitando assim múltiplos objetos a partirem erroneamente – tentando utilizar ao mesmo tempo.
Reactor	Um objeto reator fornece uma interface assíncrona para os recursos que devem ser manuseados de forma síncrona.
Read-Write Lock	Permite acesso de leitura simultânea de um objeto, mas requer acesso exclusivo para operações de gravação.
Scheduler	Controla quando segmentos podem executar código single-threaded.
Thread Pool	Um número pré-determinado de threads é criado para executar uma série de tarefas, que são normalmente organizadas em uma fila. Normalmente há mais tarefas do que threads. Pode ser considerado um caso especial do padrão pool de objeto.
Thread-Specific Storage	Memória estática ou global local de um thread.

Documentação

A documentação para um padrão de design descreve o contexto em que o padrão é usado, as forças dentro do contexto que o padrão trabalha e a reposta do trabalho do padrão. Não há um formato único para documentar padrões de projetos. Em vez disso, uma variedade de formatos tem sido usada por diferentes autores. No entanto, algumas formas de padrão tornaram-se mais conhecidos do que outras, e, consequentemente, tornam-se pontos de partida comuns para novos esforços no padrão de escrita. Um exemplo de um formato de documentação comumente

usado é o utilizado por Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides (conhecidos coletivamente como o Gang of Four, ou GoF) em seus livros, ele contém as seguintes seções:

- **Pattern Name and Classification:** um nome descritivo e único, que ajuda a identificar referindo-se ao padrão;
- **Intent:** uma descrição do objetivo por trás do padrão e a razão para usá-lo;
- **Also Known As**: outros nomes para o padrão;
- **Motivation (Forces):** um cenário composto por um problema e um contexto em que este padrão pode ser utilizado;
- **Applicability:** situações em que este padrão é utilizável; o contexto para o padrão;
- **Structure:** uma representação gráfica do padrão. Os diagramas de classe e diagramas de interação podem ser usados para essa finalidade;
- **Participants:** uma lista das classes e objetos usados no padrão e suas funções no projeto;
- **Collaboration:** uma descrição de como as classes e objetos usados no padrão podem interagir uns com os outros;
- **Consequences**: uma descrição dos resultados, efeitos colaterais, e trade-offs causados pelo uso do padrão;
- **Implementation:** a descrição de uma implementação do padrão; a parte de solução do padrão;
- **Sample Code:** uma ilustração de como o padrão pode ser usado em uma linguagem de programação;
- **Known Uses:** exemplos de usos reais do padrão;
- **Related Patterns:** outros padrões que têm alguma relação com o padrão; discussão das diferenças entre o padrão e os padrões similares.

Críticas ao uso dos padrões

O conceito de padrões de projeto tem sido criticado de diversas maneiras. Os padrões de projeto podem ser apenas um sinal da falta de recursos de determinadas linguagens de programação. Há autores que afirmam que esses padrões podem ser simplificados ou totalmente eliminados com o uso de suporte direto à linguagem. Outros que demonstram ainda que, com a utilização de Linguagens Orientadas a Objetos, a dependência de nível de código foi feita usando metade dos padrões apenas, focar nessa forma de linguagem é a saída para a facilitação da programação. Além disso, o uso inadequado de padrões pode aumentar desnecessariamente a complexidade.

Material Complementar

PRESSMAN, R. S. **Engenharia De Software.** 6ª ed. Porto Alegre: Grupo A, 2010. (e-book)

STEPHEN R. S. **Engenharia de Software.** 8ª ed. Porto Alegre: Grupo A, 2008. (e-book)

PADUA, W. **Engenharia de Software,** 3ª ed. Rio de Janeiro: Grupo GEN, 2008. (e-book)

KALINOVSKY, A. **Java Secreto:** técnicas de descompilação, patching e engenharia reversa. São Paulo: Pearson, 2009. (e-book)

PFLEEGER, S. L. **Engenharia de Software:** teoria e prática - 2º ed. São Paulo: Pearson, 2009. (e-book)

Referências

Bibliografia Fundamental

SOMMERVILLE, I. **Engenharia de Software.** 9ª ed. São Paulo: Addison-Wesley, 2007. (e-book)

Bibliografia Básica

SCHACH, S. R. **Engenharia de Software:** Os Paradigmas Clássicos & Orientado a Objetos. 7ª ed. São Paulo: Bookman, 2009.

WAZLAWICK, R. S. **Analise e Projeto de Sistemas de Informação Orientados a Objetos**. 2ª ed. Rio de Janeiro: Elsevier, 2011.

Anotações

Educação a Distância
Cruzeiro do Sul Educacional
Campus Virtual

www.cruzeirodosulvirtual.com.br
Campus Liberdade
Rua Galvão Bueno, 868
CEP 01506-000
São Paulo SP Brasil
Tel: (55 11) 3385-3000

Universidade Cruzeiro do Sul

UNICID Universidade Cidade de S. Paulo

UNIFRAN Universidade de Franca

UDF Centro Universitário

Módulo Centro Universitário