Fundação CECIERJ - **Vice Presidência de Educação Superior a Distância**

Curso de Tecnologia em Sistemas de Computação
Disciplina de Arquitetura e Projeto de Sistemas II
Gabarito - AP2 1° semestre de 2008.

Nome –

Assinatura –

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

Questão 1 (3 pontos)

O padrão "Composite" visa compor objetos utilizando uma estrutura de árvore para representar hierarquias de todo-parte.

(a) Descreva um exemplo de situação onde esse padrão é útil.

Resposta:

Um possível exemplo de situação onde esse padrão é útil é no projeto de um sistema de arquivos, onde a classe "diretório" representa o "todo" e a classe "arquivo" representa a "parte". Desta forma, alguns métodos podem se comportar de forma diferente, porém recursiva, para cada uma dessas entidades. Um exemplo é o método de cálculo do tamanho do elemento, que retorna diretamente o tamanho do arquivo para a classe "arquivo" e aplica recursão sobre os sub-elementos para a classe "diretório".

(b) Quais são os benefícios alcançados com o uso desse padrão?

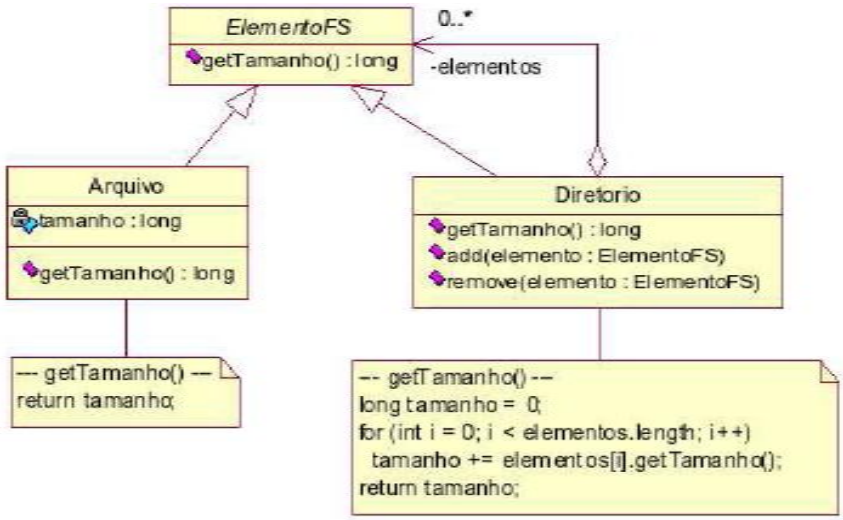
Resposta:

O principal benefício alcançado pelo uso desse padrão é permitir que tanto objetos que representam o "todo" quanto objetos que representam a "parte" possam ser tratados de maneira uniforme. Além disso, a estrutura utilizada pelo padrão permite que novas

entidades sejam adicionadas à taxonomia sem que mudanças drásticas sejam necessárias.

(c) Desenhe um modelo de classes exibindo como o padrão pode ser utilizado na situação descrita.

Resposta:



Questão 2 (2 pontos)

Em relação aos estilos arquiteturais para web Thin Client, Scripted Client, AJAX, Thick Client e Web Delivery,

(a) Descreva três desses estilos, explicando em que situação eles devem ser utilizados.

Resposta:

Thin Client - Mínima utilização dos recursos do cliente, exigindo mínima capacidade de seu navegador. Deve ser utilizado em aplicações para Internet com pouca largura de banda e sem previsão sobre arquitetura/sistema operacional/browser do cliente.

Scripted Client - Utilização de scripts de cliente apenas para a verificação de dados e dinâmica da interface com o usuário. Deve ser utilizado em um cenário equivalente ao anterior, contudo com alguma previsão sobre o browser utilizado pelo cliente (suporte a interpretação dos scripts).

AJAX - Utilização de scripts de cliente de forma assíncrona, permitindo a carga parcial de páginas. Deve ser utilizado em um cenário equivalente ao anterior, possibilitando um grau adicional de interação com o usuário devido à carga parcial de conteúdo.

Thick Client - Distribuição da lógica de negócio entre o cliente e o servidor. Aplicável principalmente para aplicações em intranet, devido ao aumento do peso da aplicação e necessidade de previsão sobre o browser utilizado.

Web Delivery - Utilização de protocolos complementares ao HTTP, tais como RMI ou COM+. Cenário semelhante ao anterior, mas com uma dependência ainda maior a características específicas da máquina cliente, o que leva a uma necessidade de alto controle sobre a configuração da mesma.

(b) Em uma situação onde a largura de banda é estreita e o cliente pode estar utilizando browser com suporte somente a HTML, qual é o estilo mais indicado? Justifique a sua resposta.

Resposta:

Neste cenário, o estilo mais indicado é o Thin Client. Como dito no enunciado, "o cliente pode estar utilizando browser com suporte somente a HTML". Com isso, nenhum dos outros estilos é aplicável, pois eles demandam a execução de scripts ou componentes no cliente, o que não é possível com browsers simples, que suportam somente HTML. Desta forma, o estilo Thin Client atenderia aos requisitos apresentados.

Questão 3 (3 pontos)

"No contexto de criação intensiva de sistemas de software, a reutilização é qualquer procedimento que produz (ou ajuda a produzir) um sistema através da reutilização de algo que foi construído num esforço de desenvolvimento prévio." A partir desta definição, responda:

(a) Quais são os possíveis benefícios trazidos pela reutilização de software?

Resposta:

Alguns de seus benefícios trazidos pela reutilização de software são: (i) melhores índices de produtividade; (ii) produtos de melhor qualidade, mais confiáveis, consistentes e padronizados; (iii) redução dos custos e tempo envolvidos no desenvolvimento de software; e (iv) maior flexibilidade na estrutura do software produzido, facilitando sua manutenção e evolução.

(b) Quais são as atuais barreiras que impedem uma implantação efetiva de uma estratégia de reutilização de software?

Resposta:

Algumas das atuais barreiras são: (i) identificação, recuperação e modificação de artefatos reutilizáveis; (ii) compreensão dos artefatos recuperados; (iii) garantia da qualidade dos artefatos reutilizáveis; (iv) composição de aplicações a partir de componentes; (v) barreiras psicológicas, legais e econômicas; e (vi) a necessidade da criação de incentivos à reutilização.

(c) Que atividades podem facilitar o estabelecimento de uma abordagem de reutilização de software?

Resposta:

Pode-se definir um Programa de Reutilização que, basicamente, estabelece uma estratégia de reutilização e um plano para sua implementação dentro da organização. Este plano geralmente inclui atividades para a criação, gerência e utilização de artefatos reutilizáveis. A primeira atividade visa produzir software e produtos associados para a reutilização; a segunda busca coletar, avaliar, descrever e organizar artefatos reutilizáveis para garantir sua disponibilização às demais atividades; e a terceira envolve técnicas para a composição de um sistema a partir de artefatos reutilizáveis.

Questão 4 (2 pontos)

O que torna arquitetura de software um tópico relevante hoje em dia? Quais são os benefícios de se considerar a arquitetura de software durante o desenvolvimento de um sistema?

Resposta:

Podemos citar uma série de justificativas sobre a relevância da arquitetura de software no cenário atual da Engenharia de Software. Uma das principais está no fato de que produtos de software são entidades que se encontram em constante estado de mudança, sobretudo no contexto de desenvolvimento atual, que contempla novas formas de interação como desenvolvimento distribuído de software. Dessa forma, com a necessidade de evolução, os produtos de software se tornam predispostos a defeitos, atrasos na entrega e custos acima do esperado. Ou seja, seu escopo, complexidade e manutenção são cada vez mais significativos e exigem que profissionais da área se comuniquem por meio de componentes de software, uma vez que percebemos que o sucesso de um sistema computacional depende muito mais das características do software produzido do que do hardware sobre o qual irá rodar. Somando-se a isso, técnicas de abstração como decomposição modular, linguagens de programação de alto nível e tipos de dados abstratos já não são mais suficientes para lidar com os

requisitos envolvidos nos domínios de aplicação atuais. Por outro lado, a arquitetura de software visa incorporar a disciplina de reutilização e agregar suas vantagens, evitando retrabalho ou trabalho desnecessário, além de permitir aos engenheiros de software tomarem decisões sobre alternativas de projeto. Enfim, o foco recente em arquitetura de software se deve, sobretudo, à economia e à reutilização.

Dentre os benefícios de se considerar a arquitetura de software durante o desenvolvimento de um sistema, estão: (i) favorecer a gerência da complexidade; (ii) facilitar a comunicação entre os stakeholders envolvidos no desenvolvimento de software (desenvolvedores, clientes, gerentes, etc.); (iii) criar possibilidades de reutilização; (iv) viabilizar a evolução de sistemas; (v) reconhecer estruturas comuns entre sistemas; (vi) manter o controle da produção de software, já que os detalhes relativos a esforços e desenvolvimento e impacto de mudanças tornam-se mais claros; e (vii) permitir novas oportunidades para análise (ex. consistência, atributos de qualidade, atendimento a estilos arquiteturais).