PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA DE CIÊNCIAS EXATAS E DA COMPUTAÇÃO.
ANÁLISE E DESENVOLVIMENTO DE SISTEMAS.

ESQUEMA COMPARATIVO SOBRE ESTILOS ARQUITETURAIS

BRUNO CAMARGO MANSO
JOÃO VICTOR CARDOSO DE OLIVEIRA
NIKOLLY CARDOSO DE FARIA

GOIÂNIA, GO
2020

BRUNO CAMARGO MANSO
JOÃO VICTOR CARDOSO DE OLIVEIRA
NIKOLLY CARDOSO DE FARIA

ESQUEMA COMPARATIVO SOBRE ESTILOS ARQUITETURAIS

Esquema comparativo de todas características, vantagens e desvantagens dos principais estilos arquiteturais

Desenho de Software
Orientador: Aníbal Vicente Vieira

GOIÂNIA, GO
2020

Resumo

No presente trabalho, iremos explorar as características de seis estilos arquiteturais esquematizando-os em uma tabela facilitando assim, o estudo comparativo entre eles. Os estilos serão devidamente descritos bem como suas vantagens e desvantagens.

Palavras-chave: Estilos arquiteturais, comparativo, descrição, vantagens, desvantagens

Sumário

1. Introdução

Este trabalho tem o objetivo de fazer um esquema comparativo entre 6 estilos arquiteturais mais usados. Demonstraremos em tabela as características, vantagens e desvantagens de cada estilo. O trabalho fora elaborado a partir de diversas fontes escolhidas pelos alunos.

2. Esquema comparativo de todas características

Vantagens e desvantagens dos principais estilos arquiteturais.

	Características	Vantagens	Desvantagens
Client-server	- É uma arquitetura na qual o processamento da informação é dividido em módulos ou processos distintos. - Existe um processo que é responsável pela manutenção da informação (servidores) e outro responsável pela obtenção dos dados (os clientes). - Cliente: Solicita um determinado serviço através do envio de uma "mensagem"(pode ser chamado também de solicitação) ao servidor. Servidor: Executa as tarefas solicitadas pelo cliente e enviam uma resposta.	- Recursos centralizados -Maior facilidade de manutenção -Front-end e Back-end separados -Maior segurança pois o servidor pode conter suas próprias regras de segurança além das regras do cliente -Funciona com vários clientes diferentes com capacidades diferentes.	-Sobrecarga -Único nó -Cliente podem solicitar serviços, mas não podem oferecer para outros clientes.
Arquitetura baseada em Componentes	-O desenvolvimento baseado em componentes permite que o sistema final seja tratado como vários "mini-sistemas", diminuindo sua complexidade e permitindo que cada componente empregado seja	-Produtividade: Pode-se economizar tempo de desenvolvimento, dependendo do portfólio de componentes já prontos -Robustez: Maior qualidade	-Mudanças nem sempre são bem vindas por contar com uma série de elementos muitas vezes deslocados -Muitos veem como "se funciona para que mecher"

	focado em apenas uma funcionalidade ou um conjunto de funcionalidades semelhantes -Componentes podem ter dependências com outros componentes. Neste caso um componente realiza a chamada para a interface de outro componente.	no produto final que utiliza componentes, pois os mesmos já foram largamente testados em um projeto dedicado à construção dos mesmos; -Padrão de desenvolvimento: Equipe orientada a desenvolvimento nos moldes da componentização.	por conta de contar também com elementos desacoplados.
Arquitetura em Camadas	- É um processo de decomposição de sistemas complexos em camadas para facilitar a compreensão do mesmo e facilitar a manutenção deste sistema. - Fora emprestada da arquitetura de computadores que utilizam chamadas de camada ao sistema operacional, drivers etc. - Organiza o sistema em um conjunto de camadas em outras palavras: máquinas abstratas. - Cada camada fornece um conjunto de serviços e cada camada é cliente da camada subjacente. - Existe a generalização do cliente-servidor além de não precisar ser distribuído. - Apoia o desenvolvimento incremental dos subsistemas, em camadas diferentes. Ao por exemplo se mudar a camada de	- Melhor compreensão - Facilidade de manutenção - Desenvolvimento independente - Isola as funções do sistema operacional facilitando sua manutenção e depuração - Cria hierarquia de níveis de modos de acesso, protegendo camadas mais internas.	- Duplicação de funcionalidade - Dificuldade de estruturação de um sistema por camadas. - Facilidade de violação da estrutura vigente -Overhead de implementação e desempenho - Baixo desempenho - Uma nova camada implica em uma mudança de modo de acesso - Sistemas comerciais utilizam apenas 2 camadas onde existem modo de acesso pelo usuário de forma não privilegiada e de modo privilegiado via kernel somente.

	negócios,apenas as camadas acima precisarão de modificação. **Camada de apresentação:** - Responsável pela apresentação e interação homem máquina com recebimento de dados processados pelo software e suas funções. **Camada de Controle:** - Comanda o fluxo da apresentação servindo como uma camada intermediária entre a camada de apresentação e a lógica **Camada lógica de negócios:** - É a implementação das regras e requisitos de negócio **Camada de Acesso a Dados:** - Possui técnicas de persistência de dados. - Encapsula e expõe os dados. - Geralmente persistido em bancos de dados **Injeção de dependências:** **-** A responsabilidade para a apresentação e interação do usuário reside nos componentes da primeira camada. Esses componentes clientes permitem ao usuário interagir com os processos da segunda camada de uma		

	maneira segura e intuitiva.		
Orientado a Objetos	- Conjunto de objetos fracamente acoplados e de interface bem definida. - Objetos interagem uns com os outros através da invocação de funções ou procedimentos (métodos) - Objetos são responsáveis por manter sua integridade - Sua representação é oculta aos outros objetos - Baseado na composição e interação entre diversas unidades chamadas de objetos - Objetos oferecem um conjunto de serviços - Empregados em sistemas distribuídos em forma de Objetos distribuídos - A implementação orientada a objetos não implica em arquitetura de objetos	- Objetos fracamente acoplados devido ao uso de interfaces - Ampla utilização de implementações orientadas a objetos nos dias de hoje - Necessidade de comunicação gera dependência - A alteração de um objeto pode afetar todos os demais que dependem dele, pode gerar efeito em cascata	- Mudanças na interface sempre com alto impacto - Dificuldade de manutenção devido a falta de restrições topológicas - limitações sobre a dependência entre objetos
Service-Oriented Architecture (SOA)	-Conceito de arquitetura corporativo que promove a integração entre o negócio e a TI por meio de conjunto de interfaces de serviços acoplados. -Modelo de planejamento de estratégia que se alinha diretamente aos objetivos de negócios de uma organização.	**-Reutilização**: O serviço pode ser reutilizado para outras aplicações. **-Produtividade**: Com o reuso, a equipe de desenvolvimento pode reutilizar serviços em outros projetos, diminuindo o tempo de desenvolvimento.	-Em uma pesquisa global patrocinada pela CA, 43% dos executivos classifica a segurança como o ponto mais crítico nas iniciativas SOA; **-Complexidade**: Uma grande quantidade de serviços precisa ser gerenciada.

	-Tem como objetivo integrar as aplicações, disponibilizar maior flexibilidade para mudanças, suportar serviços independentes de plataforma e protocolos. -Trata os requisitos de baixo acoplamento, desenvolvimento baseado em padrões, computação distribuída independente de protocolo, integração de aplicações e sistemas legados. -Tem como componente o ESB, que é um software de infraestrutura que torna os serviços de negócios reutilizáveis e amplamente disponíveis para usuários, aplicações, processo e outros serviços. -Traz maiores prioridades de inovação, aumentando a capacidade de colaboração entres aplicativos e inovando os modelos de negócio e processos. -O foco em SOA é a construção e disponibilização de serviços de negócio, evitar replicação de dados, reuso e facilidade de manutenção de sistemas, integração entre os sistemas, visão e controle do processo de negócio, agilidade nas mudanças.	**-Flexibilidade**: Isolando a estrutura de um serviço as mudanças são feitas com maior facilidade. **-Manutenibilidade**: Com baixo acoplamento, facilita a manutenção dos serviços. **-Alinhamento com o negócio**: A área de negócio visualiza os processos alinhados com a tecnologia. **-Interoperabilidade**: Disponibilizar serviços independentemente da plataforma e tecnologia. **-Integração**: A integração com outros serviços, aplicativos e sistemas legados. **-Governança**: Gerenciamento nos processamentos de negócio. **-Padronizado**: É baseado no uso de padrões. **-Abstração**: Serviço totalmente abstraído da sua implementação.	**-Performance**: A performance depende do servidor onde o serviço está publicado, como também da rede. **-Robustez**: Caso uma exceção acontecer não tem como reverter o processo. **-Disponibilidade**: Uma queda na rede ou no servidor deixa todos os serviços indisponíveis. **-Testabilidade**: O debug no serviço é um problema para os desenvolvedores. **-Segurança**: Os serviços estão disponíveis na rede, qualquer aplicativo pode consumir esse serviço, os dados são trafegados pela rede podendo ser interceptados.

Filtros e dutos (pipes and filters)	-Originário de sistemas operacionais UNIX e do projeto de compiladores -Transformações funcionais processam entradas para produzir saídas. – Componentes são chamados de filtros -Conectores são dutos (pipes) -Útil para aplicações de processamento de informação que interagem pouco com usuários -Variação distribuída: processamento de streams.	-Apóia reuso de transformações. -É fácil adicionar novas transformações. -É relativamente simples implementar como sistema concorrente ou seqüencial.	-Requer um formato comum para a transferência de dados ao longo do pipeline -Não é apropriado para aplicações interativas

3. Bibliografia

Aleixo, Felipe. Estilos Arquiteturais. **Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte**, 2020. Disponível em: http://tinyurl.com/yyzfz4tj. Acesso em: 8 de outubro de 2020.

Arquitetura em Multicamada. **Wikipédia**, 2020. Disponível em: http://tinyurl.com/y2sqnrr3. Acesso em: 8 de outubro de 2020.

Cornélio, Marcio. Estilos Arquiteturais. **Centro de Informática da Universidade Federal de Pernambuco**, 2020. Disponível em: http://tinyurl.com/y6ahstew. Acesso em: 8 de outubro de 2020.

Vantagens e Desvantagens de SOA. **DevMedia**, 2020. Disponível em: http://tinyurl.com/yxfwlven. Acesso em: 8 de outubro de 2020.

Arquitetura em Multicamada. **Wikipédia**, 2020. Disponível em: http://tinyurl.com/y3h684p9 Acesso em: 8 de outubro de 2020.

Desenvolvimento Baseado em Camadas. **DevMedia,** 2020. Disponível em: http://tinyurl.com/yyk7def2 Acesso em: 8 de outubro de 2020.