RECOMPOSIÇÃO DE CARGA HORÁRIA – 09/12/2023
REVISÃO - TÓPICOS AVANÇADOS EM ENGENHARIA DE SOFTWARE

INSTRUÇÕES

Com base nos conhecimentos apresentados na aula, e pesquisas relacionadas, realize os exercícios a seguir:

Após, encaminhe suas as respostas em um arquivo no **Classroom**.

DATA DE ENTREGA: **até 14/12/2023**

EXERCÍCIOS

Engenharia de Software Distribuído.

A) O que você entende por **'escalabilidade'**? Discuta as diferenças entre **escalar verticalmente** e **escalar horizontalmente**, e explique em quais casos podem ser utilizada essas diferentes abordagens para a escalabilidade *(Valor 10,0)*.

É quando o sistema consegue lidar com um aumento na demanda sem perder a qualidade e o desempenho do processo.

A Escalabilidade Vertical é aumentar a capacidade de um único componente, como adicionar mais recursos (CPU, RAM, disco) a um servidor existente, pode ser usada quando os requisitos podem ser atendidos por uma única máquina poderosa ou quando o aumento de carga é previsível, e o hardware mais poderoso é suficiente para lidar com isso.

A Escalabilidade Horizontalmente é adicionar mais instâncias ao sistema, distribuindo a carga entre várias máquinas, pode ser usada quando a aplicação pode ser dividida em partes independentes ou quando o crescimento é imprevisível, e ter mais máquinas é uma solução mais flexível.

B) Explique por que os **Sistemas de Software Distribuídos** são mais complexos do que os **Sistemas de Software Centralizado**? *(Valor 10,0)*

Porque um sistema centralizado tem um único ponto de falha, a comunicação entre componentes geralmente ocorre internamente, na mesma máquina. Enquanto um sistema distribuído não tem um único ponto de falha, pois Componentes em sistemas distribuídos frequentemente residem em máquinas diferentes e se comunicam através de redes
Isso acaba deixando o sistema distribuído mais complexo, pois tem que se preocupar com a latência de rede, perda de pacotes e gerenciamento de conexões.

C) Qual é a diferença fundamental entre uma abordagem de cliente magro e uma de cliente gordo para as arquiteturas de sistemas **cliente-servidor**? *(Valor 10,0)*

No Modelo Cliente-Magro, todo processamento da aplicação e o gerenciamento de dados é realizado no servidor. O cliente é responsável somente por executar o software de apresentação. No Modelo Cliente-Gordo, o servidor somente é responsável pelo gerenciamento de dados e o software do cliente implementa a lógica da aplicação e as interações com os usuários.

D) Sobre a abordagem de **Software como Serviço**, como esse modelo pode reduzir os custos de suporte? E quais custos adicionais poderiam surgir? *(Valor 20,0)*

Pode reduzir pois possui acesso remoto e escalabilidade, isso pode reduzir a necessidade de visitas presenciais, economizando tempo e custos associados. O SaaS pode lidar com escalabilidade, o que significa que as soluções podem se adaptar à demanda sem aumentar os custos de suporte.

Mas podem surgir custos associados à assinatura recorrente, personalizações específicas que podem ser necessárias e requisitos específicos da organização.

Engenharia de Software Orientada a Serviços

A) Quais são as distinções mais importantes entre os ***Serviços*** e os ***Componentes de Software***? *(Valor 10,0)*.

- Serviços devem ser mais fracamente acoplados do que os componentes
- Os componentes podem ser instalados fisicamente na máquina de seus clientes/consumidores enquanto os serviços são acessados apenas remotamente.
- Em serviços é enfatizado a construção de sistemas distribuídos, flexíveis e interoperáveis
- Os componentes de software podem ter uma variedade de abordagens de design, dependendo do contexto e das necessidades específicas do sistema.

B) O que é Arquitetura Orientada a Serviços (**SOA**)? *(Valor 10,0)*.

É um método de desenvolvimento de software que usa componentes de software chamados de serviços para criar aplicações de negócios. Cada serviço fornece um recurso de negócios, e todos eles também podem se comunicar entre si em diferentes plataformas e linguagens.

C) Quais as principais diferenças entre **SOAP** e **REST**? *(Valor 10,0)*.

SOAP	REST

Design	A API SOAP expõe a operação.	A API REST expõe os dados.
Protocolo de transporte	O SOAP é independente e pode funcionar com qualquer protocolo de transporte.	O REST funciona somente com HTTPS.
Formato dos dados	O SOAP oferece suporte somente para a troca de dados XML.	O REST oferece suporte XML, JSON, texto simples e HTML.
Performance	As mensagens SOAP são maiores, o que torna a comunicação mais lenta.	O REST tem uma performance mais rápida devido às mensagens menores e ao suporte para armazenamento em cache.
Escalabilidade	O SOAP é difícil de escalar. O servidor mantém o estado armazenando todas as mensagens anteriores trocadas com um cliente.	O REST é fácil de escalar. Como ele não tem estado, cada mensagem é processada independentemente das anteriores.
Segurança	O SOAP oferece suporte para criptografia com sobrecargas adicionais.	O REST oferece suporte para criptografia sem afetar a performance.
Caso de uso	O SOAP é útil em aplicações legadas e APIs privadas.	O REST é útil em aplicações modernas e APIs públicas.

D) Pesquisando a respeito de ***Web Services*** e ***APIs***, defina o que são, no que são semelhantes e no que diferem. *(Valor 20,0)*.

API é uma interface que conecta dois programas, realizando a comunicação entre eles e especificando como seus softwares devem interagir.

Já os web services são APIs que se comunicam por meio de redes e podem ser combinados para a execução de operações complexas.

Semelhanças:

- Ambos facilitam a comunicação e integração entre sistemas de software
- Ambos usam padrões de comunicação, como SOAP ou REST.
- Ambos permitem que desenvolvedores acessem funcionalidades ou dados oferecidos por um serviço ou aplicativo.

- Ambos usam protocolos HTTP.

Diferenças:

- Web service é uma aplicação enquanto a API facilita a interface direta com um aplicativo.
- Nem todas as APIs são Web services, porém, todos os Web Services são APIs.
- Web Services não executam todas as tarefas realizadas ou não de uma API.
- A API pode utilizar qualquer estilo de comunicação, porém o serviço Web só executa apenas três estilos de comunicação que são eles SOAP, REST e XML-RPC.
- A API não precisa de uma rede para seu funcionamento acontecer, enquanto o Web Server depende disto.