ES3A3 – Exercícios

Aula 1 – Visão Geral do Projeto de Software

1. Quais as diferenças entre o modelo de requisitos (ou de análise) de software e o modelo de
projeto de software?

R: O modelo de projeto de software fornece detalhes sobre a arquitetura da aplicação, ou seja, a forma como o software será construído; enquanto o modelo de requisitos fornece uma forma de agrupar interesses, na descrição de funções, dados e etc.

2. Descreva brevemente como é feito o projeto de software nos processos de desenvolvimento
em cascata, espiral, RUP (Rational Unified Process) e XP (Extreme Programming).

1 : Cascata - > É um modelo sequencial de desenvolvimento de software, cada passo é desenvolvido de forma linear. Cada etapa deve ser concluída para que a próxima possa iniciar.

2 : Espiral -> É um modelo de iterativo de desenvolvimento de software, tendo maior flexibilidade em comparação ao modelo cascata. Cada etapa é feita através de ciclos que passam pelas mesmas fases do modelo em cascata. Além disso, esse modelo não limita o início de uma fase enquanto a anterior não for concluída, além de permitir modificações ao longo do desenvolvimento.

3 : RUP -> É um modelo iterativo, que se baseia em um ciclo de 4 fases para a construção de um software, sendo elas : concepção, elaboração, construção e transição.

4 : XP -> O XP é um modelo de desenvolvimento de software ágil que enfatiza a entrega contínua de software funcionando em ciclos curtos.

3. Explique com suas próprias palavras como é feita a transformação do modelo de requisitos de
software para o modelo de projeto de software.

A conversão do modelo de requisitos em modelo de projeto é uma etapa fundamental, pois é nela que os insumos fundamentais do modelo de requisito são transformados em informações mais detalhadas, que posteriormente servirão de base para a etapa de codificação.

4. Padrões de arquitetura e padrões de projeto são a mesma coisa? Explique.

R: Padrões de arquitetura são estruturas de alto nível que orientam a organização de sistemas de software complexos, enquanto padrões de projeto são soluções específicas para problemas de design recorrentes em nível de código. Ambos são ferramentas importantes para garantir a qualidade e a eficiência de um sistema de software.

5. Qual a relação entre o conceito de separação por interesses e a expressão popular “uma coisa
de cada vez”?

Tanto o conceito de separação por interesses na programação de software quanto a expressão popular "uma coisa de cada vez" enfatizam a importância de manter o foco em uma única tarefa ou responsabilidade de cada vez para maximizar a eficiência e o sucesso.

6. Qual a relação entre os conceitos de separação por interesses, modularidade, encapsulamento
de algoritmos e dados e aspectos?

Os conceitos de separação por interesses, modularidade, encapsulamento de algoritmos e dados e aspectos estão relacionados na programação de software, pois todos eles visam a promover a organização e a estruturação

do código em módulos independentes e coesos, o que facilita a manutenção, evolução e reutilização do software.

7. Em que situação a modularização do software pode começar a se tornar cara? Existe um nível
de modularização satisfatório? Se sim, como determiná-lo?

R : A modularização do software pode começar a se tornar cara quando o número de módulos se torna excessivo e a complexidade da interação entre eles aumenta. Isso pode levar a problemas de desempenho, como aumento no tempo de compilação e execução do programa, e pode dificultar a manutenção do código.

8. Como o conceito de independência funcional contribui para a qualidade de um projeto de
software?

O conceito de independência funcional contribui para a qualidade de um projeto de software, pois promove a separação de responsabilidades e a redução do acoplamento entre os componentes do sistema, o que facilita a manutenção, evolução e reutilização do código.