**Testes**

**1 – Diferença entre validação e verificação?**

Verificação – SW = especificação, deve evidenciar defeitos.

Validação – SW = necessidades do usuário, opera como planejado.

**2 – Vantagens inspeções sobre testes.**

Inspeções – verificação estática, documentos e análise de dados (especificações, código-fonte, UML, banco de dados).

Testes – execução do sistema e observação do comportamento (protótipo e programa).

A inspeção é mais detalhada, encontra erros que podem não ser encontrados nos testes, pode ser em versões incompletas facilmente, não corrige apenas defeitos, mas pode melhorar a arquitetura do programa.

**3 – Breve descrição dos três principais estágios e testes para um SW comercial**

Testes de desenvolvimento – o sistema é testado durante seu desenvolvimento (pela equipe de desenvolvimento) para descobrir bugs e defeitos. Testes de unidade/componentes/sistema.

Testes de release – uma equipe de testes testa uma versão completa do sistema antes que ele seja liberado para usuários (alpha test).

Testes de usuário – usuários testam o sistema em seu próprio ambiente (beta).

**4 – Que testes devem ser incluídos no teste de classe de objeto?**

Testes de todas as operações associadas a um objeto, definição e verificação de todos os atributos do objeto e exercitar o objeto em todos os estados possíveis.

**5 – Qual guia Whittaker sugere para detectar testes?**

Escolha entradas que forcem o sistema e gerar todas as mensagens de erro.

Desenvolva entradas que causem overflow nos buffers de entrada.

Repita a mesma entrada ou série de entradas várias vezes.

Force a geração de saídas inválidas.

Force resultados de computação a serem grandes ou pequenos demais.

**6 – O que é uma partição de equivalência? Exemplos.**

Uma classe onde todos os membros estão relacionados, onde o programa se comporta de uma forma equivalente para cada membro da classe. Por exemplo, para se acessar um site, deve-se

digitar a idade. Qualquer valor negativo é descartado, entre zero e 17 o site fecha e acima de 18 o acesso é permitido. Pra um conjunto de valores, o sistema se comporta igual. Um conjunto de entradas ou saídas tem a mesma resposta do sistema.

**7 – Quais são as três classes importantes de erros de interface?**

Mau uso de interface – um componente chama outro e comete um erro no uso da sua interface.

Mau entendimento de interface – um componente chamador incorpora suposições incorretas sobre o comportamento dos componentes chamados.

Erros de timing – chamador e chamado operam em velocidades diferentes e são acessadas informações desatualizadas.

**8 – Quais devem ser as principais preocupações dos testes de um sistema?**

Integração dos componentes para criar uma versão do sistema e testar.

Testar interações entre componentes.

Ver se os componentes são compatíveis, interagem corretamente e transferem os dados certos no momento certo por suas interfaces.

Testam o comportamento emergente de um sistema.

**9 – Descreva brevemente o processo de desenvolvimento dirigido a testes.**

Testes são escritos antes do código-fonte e o objetivo do desenvolvimento é passar nestes testes. O desenvolvimento é incremental e o próximo incremento só é feito após o anterior passar no teste. Pensa em um incremento (mas não faz) > escreve um teste pra ele > executa todos os testes > implementa o incremento. Se passar, vai pro próximo, se não, refatora (óbvio que o problema está no incremento) e testa de novo.

**10 – O que é teste de cenário?**

O teste de cenário testa comportamento do sistema em diferentes cenários para saber se ele se comporta como deveria. Ex: se autentica login, se aparece a mensagem de erro, etc.

**11 – O que é teste de stress e por que ele é útil?**

Deliberadamente sobrecarregar o sistema para testar seu comportamento até falhar.

**12 – Quais são os três tipos de testes de usuários?**

Alfa – usuários trabalham com os desenvolvedores para testar o SW no local do

desenvolvedor.

Beta – uma versão é disponibilizada para que usuários possam testar em suas casas e os problemas encontrados são relatados aos desenvolvedores.

Aceitação – clientes testam um sistema e decidem se ele está pronto para ser implantado no ambiente do cliente. Principalmente in demand.

**Evolução**

**1 – Por que a evolução de um SW é tão importante?**

Sempre surgem novos requisitos, mudanças no ambiente, ocorrem erros que devem ser reparados, equipamentos novos são disponibilizados e o sistema deve se adequar a ele, o desempenho e a confiabilidade do sistema deve sempre melhorar. Os SW são propriedade da empresa e possuem valor ativo, então devem ser alterados e mantidos atualizados.

**2 – Quais são os estágios no processo de evolução de um sistema e o que inicia o processo?**

Processo de Identificação de Mudanças > Proposta de mudança > Processo de evolução de SW > Novo Sistema. No processo de evolução: solicitação de mudança – análise de impacto – planejamento de release – implementação de mudança – liberação do sistema. Propostas de mudança engatilham.

**3 – Por que algumas vezes pode ser necessário ignorar o sistema de gestão de mudanças normal e fazer mudanças urgentes em um sistema?**

Defeito grave que precisa ser reparado, mudança no ambiente que tem efeitos inesperados, mudanças de negócio que exijam uma mudança rápida.

**4 – O que são as Leis de Lehman e como elas foram derivadas?**

São observações que se aplicam a sistemas de grandes corporações. Basicamente, dizem que um SW deve estar sempre mudando para se adequar ao ambiente e às necessidades do usuário ou a qualidade do sistema cairá, entre outras coisas.

**5 – Quais são os três diferentes tipos de manutenção de software e como o esforço é distribuído entre estes tipos?**

Correção de defeitos, adaptação a um novo ambiente e adição ou modificação de funcionalidades do sistema. Os gastos são distribuídos em 1/6, 1/6, 2/3.

**6 – Quais fatores devem ser avaliados para entender a relação entre um sistema e seu ambiente?**

Número e complexidade das interfaces de sistema, número de requisitos de sistema inerentemente voláteis e processos de negócio nos quais o sistema é usado.

**7 – Qual processo de métrica deve ser usado para avaliar manutenibilidade?**

Número de solicitações de manutenção corretiva, tempo médio necessário para análise de impacto, tempo médio necessário para implementar uma solicitação de mudança, número de solicitações de mudança pendentes.

**8 – Quais são as principais atividades de sistemas de reengenharia?**

Tradução do código-fonte (converter nova linguagem), engenharia reversa (análise do programa para compreensão), melhoria de estrutura do programa (facilitar compreensão), modularização de programa (reorganização da estrutura do programa), reengenharia de dados (limpeza e reestruturação de dados).

**9 – Quais são as opções estratégicas para evolução de sistemas legado?**

Deve-se considerar o valor de negócio e a qualidade do sistema legado. Se ambas forem baixas, deve-se descartar, se só o valor de negócio é alto, deve-se fazer reengenharia para melhorar a qualidades, se só a qualidade é alta, substitui, descarta ou mantém, se ambos são altos, continua normalmente.